**Test Summary Report Template**

Table of Contents

[1. Introduction 3](#_Toc186449260)

[1.1 Background 3](#_Toc186449261)

[1.2 Structure of the Report 3](#_Toc186449262)

[1.3 References 3](#_Toc186449263)

[2. Overview 3](#_Toc186449264)

[3. Adjustments 4](#_Toc186449265)

[4. Assessment 4](#_Toc186449266)

[5. Results 4](#_Toc186449267)

[6. Evaluation 4](#_Toc186449268)

[7. Summary of Activities 5](#_Toc186449269)

# Introduction

## 1.1 Background

This document provides the Test Summary for the testing activities.

This project’s goal is to optimize delivery patterns, through truck diversion, and shipment, there are a total of 3 trucks that split up around the ‘map’/city. Each truck may be diverted to a package using the shortest path. In terms of testing this project, key tests are of the valid functions, which ensure a valid package is passed (e.g a valid shipment size, weight and address) as well as if a truck has space. These helper functions were rigirously tested to ensure a valid function (input wise).

## 1.2 Structure of the Report

This report is structured in the following manner:

* Section 2, Overview, provides an overview of the significant events and activities.
* Section 3, Adjustments, records any modifications of the items from those areas agreed on previously, especially in areas that may cause concern to the group accepting the test results, including any references to supporting documentation that covers the reasons for the changes.
* Section 4, Assessment, provides a brief assessment of the comprehensiveness of the testing process.
* Section 5, Results, provides a summary of the results.
* Section 6, Evaluation, provides an overall evaluation of the testing process including any observed problems and/or limitations.
* Section 7 provides a summary of the major testing activities and events.

## 1.3 References

*A list of documents referenced within this Test Summary Report document.*

* Black box Tests Document (blackbox-test-description-g3)
* White box Tests Document (whitebox-test-cases-g3)
* Integration Tests Document (integration-test-cases-g3)
* Acceptance Tests Document (acceptance-test-cases-g3)
* All scrum reports from Milestones 1 – 6
* Test Plan Document (test-plan-g3)
* dataStruct.h
* functionImplementation.c
* mapping.h / mapping.c
* main.c

# Overview

This section provides a high-level overview of the significant events and activities.

Overview:  
  
The schedules of the creation and execution of tests:

* Milestone 3: Blackbox tests were written
* Milestone 4: All blackbox tests were implemented and executed with results, whitebox tests were written
* Milestone 5 & 6: All whitebox tests were implemented and executed with results, all integration tests were written, implemented, and executed with results, and acceptance tests were written
* Milestone 6: Acceptance tests were written, implemented, and executed

Overall, the tests done were a high amount, blackbox tests were done before the code, implemented and were succesful, the desired output was achieved. This same logic applied for whitebox testing, as we already knew the code. Integration testing followed a similar logic, it integrated two functions together (such as valid and validShip) test if both suceeded. With acceptance testing, it was also done, but some test cases failed. Lastly, the type of testing which was not done consisted of regression testing, load/stress testing, but these didn’t seem that nessecary. The type of software stayed the same, which was VisulStudios Test Explorer, as well as on Windows. The data consisted of structs mainly, which were self-implemented, such as Trucks, Shipments and etc.

# Adjustments

This section is used to record any changes of the items from those areas agreed on previously, especially in areas that may cause concern to the group accepting the test results, including any references to supporting documentation that covers the reasons for the changes.

Throughout milestones 1-3, there were no necessary adjustments to either of the source code and the blackbox tests, but upon the start of milestone 4, the data structure for the implemented custom functions and the functions themselves were refined to accept the correct parameters as per the requirements as well as to streamline the process of testing those functions. There were a lot of minor improvements afterwards, but all of them were made to refine the logic of our code and make it easier to understand and fix.

During milestone 4 and 5, after the reporting of milestones 1-3, we split the validShip() function into validShip() and valid(), in which validShip() checks the weight and size of the shipment, while valid() checks if the shipment is located at a valid address in the 25x25 grid map. findNextAvailableSlot() was also added to break down the code into smaller, more manageable functions, and this new function checks the appropriate truck to see if it has any available slots for the given shipment to fit in. At the near-end of milestone 5, the function takeUserInput() was also added in support of the enhanced main.c module, which took care of the user’s input and passed it to all the necessary functions if deemed valid. The module main.c was enhanced to include the display of input and output that the user will be seeing once they use the program, based on the project manual’s display formatting.

As for any adjustments in any of the test cases, none were modified in either logic or expected results. The only changes that were made were fixes to grammar/spelling.

Documents referenced:

* dataStructs.h
* functionImplementation.c
* main.c
* Black box Tests Document (blackbox-test-description-g3)
* White box Tests Document (whitebox-test-cases-g3)
* Integration Tests Document (integration-test-cases-g3)
* Acceptance Tests Document (acceptance-test-cases-g3)

# Assessment

This section provides a brief assessment of the comprehensiveness of the testing process for the completed testing phase against the test objectives and constraints specified in the Test Plan document.

Where code coverage measurements have been made, the results should also be included in this section. This section also identifies any aspects of the AUT that were not tested as thoroughly as planned (due to insufficient time or resources).

According to the Test Plan (Test Strategy, part 3 of the Test Plan):

* Purpose of the Test: Fix and improve the given source code and ensure that the program runs and works correctly.
* Phases of Testing: Unit Testing, Functional Testing, Integration Testing, User Acceptance Testing
* Scope of Testing: Correct Shipment Distribution, Truck’s Path and Capacity Calculation, and Output Message Functions
* Critical Success Factors: When 95% of tests pass with no high or critical errors that could impact the overall program, when the map/grid is navigated correctly, and when the actual outcome matches the expected outcome in the Project Overview. It should also pass the if-statements according to the milestone assumptions.

FINAL ASSESSMENT:

At the end of milestone 6, we have successfully determined and covered the 8 requirements of the overall project. This includes:

1. Max Weight on Truck
2. Shipment Weight
3. Shipment Size
4. Getting the Shortest Path
5. Shipment Address
6. Functional Divert Process
7. Functional Truck Route
8. Functional Display/Output

All black box tests, white box tests, and integration tests have been executed properly and yielded successful (PASSED) results. Each test case in every test set covered at least one or more requirements mentioned above. All of these tests have been made and done on time for all milestones.

However, some Acceptance Test results failed due to some bugs found in the custom functions, which we were not able to further debug because of insufficient time and lack of member availability. The biggest challenge here was trying to format the display/output features correctly, since we found this particularly difficult to fix during acceptance testing.

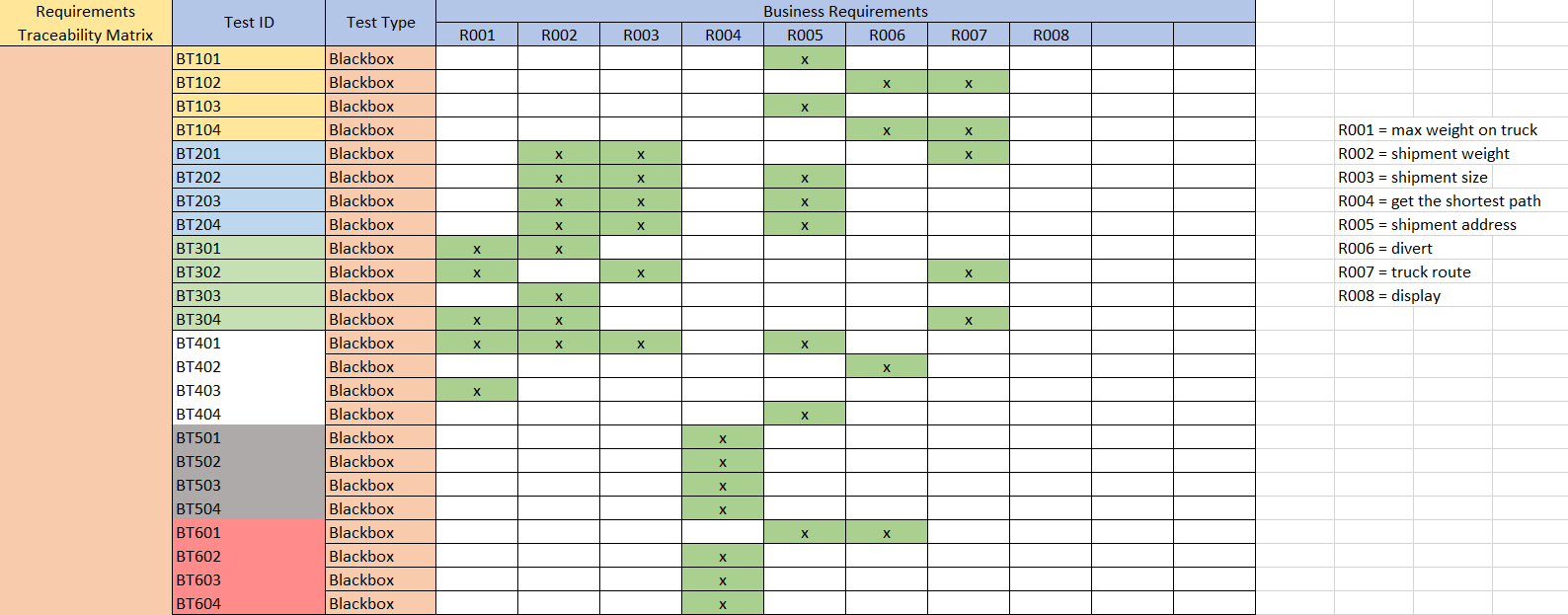
Overall, we managed to accomplish most of the objectives highlighted in the Test Plan, all except for the failed acceptance test results and the display/output issues from the program.

# Results

This section provides a summary of the results, identifies all resolved issues and summarises the details of their resolution, and lists any outstanding issues.

Requirements Traceability Matrix (Test Coverage):

BLACK BOX TEST CASES RESULTS: (based on blackbox-test-description-g3)



For black box test cases, we conducted 6 test sets:

* BT101 – BT104 tests the assignPackage() function, which covers the requirements for Shipment Address, Functional Divert Process, and Functional Truck Route. All test cases have passed. (Conducted by Bilal)
* BT201 – BT204 tests the valid() function, which covers the requirements for Shipment Weight, Shipment Size, Shipment Address, and Functional Truck Route. All test cases have passed. (Conducted by Bilal)
* BT301 – BT304 tests the checkSpace() function, which covers the requirements for Max Weight on Truck, Shipment Weight, Shipment Size, and Functional Truck Route. All test cases have passed. (Conducted by Bilal)
* BT401 – BT404 tests the divert() function, which covers the requirements for Max Weight on Truck, Shipment Weight, Shipment Address, and Functional Divert Process. All test cases have passed. (Conducted by Kaitlyn)
* BT501 – BT504 tests the mapping function eqPt(), which covers the requirements for Getting the Shortest Path. All test cases have passed. (Conducted by Kaitlyn)
* BT601 – BT604 tests the mapping function distance(), which covers the requirements for Getting the Shortest Path, Shipment Address, and Functional Divert Process. All test cases have passed. (Conducted by Kaitlyn)

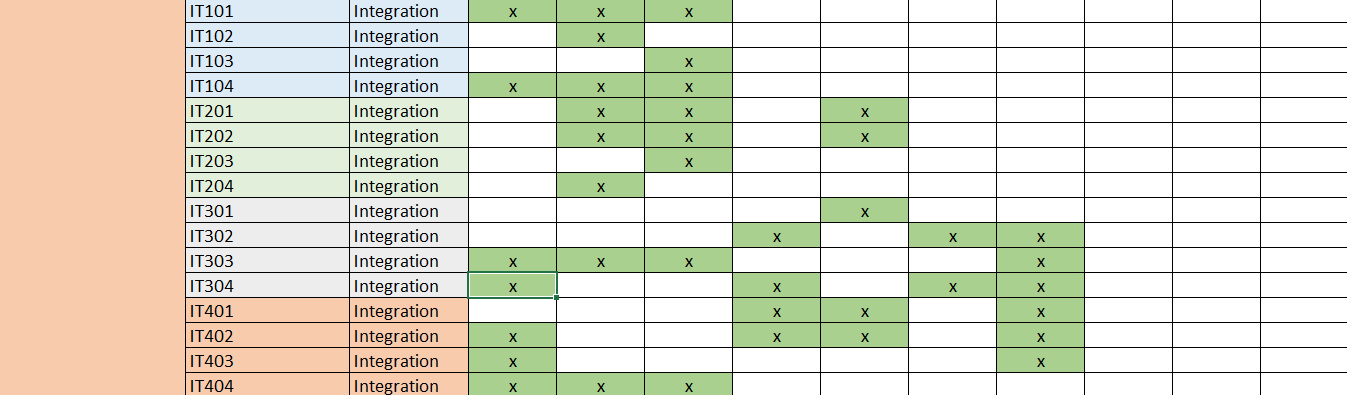
WHITE BOX TEST CASES RESULTS: (based on whitebox-test-cases-g3)

INSERT MATRIX PHOTO OF WHITE BOX HERE

For white box test cases, we conducted \_\_ test sets:

* WT101 – WT104 tests the checkSpace() function, which covers the requirements for Max Weight on Truck, Shipment Weight, Shipment Size, and Functional Truck Route. All test cases have passed. (Conducted by Kaitlyn)
* WT201 – WT204 tests the valid() function, which covers the requirements for Max Weight on Truck, Shipment Weight, Shipment Size, and Shipment Address. All test cases have passed. (Conducted by Kaitlyn)
* WT301 – WT304 tests the assignPackage() function, which covers the requirements for Max Weight on Truck, Getting the Shortest Path, Shipment Address, and Functional Truck Route. All test cases have passed. (Conducted by Grace)
* WT401 – WT404 tests the assignPackage() function, but this time it covers the requirements for Max Weight on Truck, Shipment Weight, Shipment Size, and Functional Truck Route. All test cases have passed. (Conducted by Grace)
* Add more test cases here if possible

INTEGRATION TEST CASES RESULTS: (based on integration-test-cases-g3)



For integration test cases, we conducted 4 test sets:

* IT101 – IT104 tests both checkSpace() and validShip() functions, which cover the requirements for Max Weight on Truck, Shipment Weight, and Shipment Size. All test cases have passed. (Conducted by Kaitlyn)
* IT201 – IT204 tests both validShip() and valid() functions, which cover the requirements for Shipment Weight, Shipment Size, and Shipment Address. All test cases have passed. (Conducted by Kaitlyn)
* IT301 – IT304 tests both assignPackage() and divert() functions, which cover the requirements for Max Weight on Truck, Shipment Weight, Shipment Size, Getting the Shortest Path, Shipment Address, Functional Divert Process, and Functional Truck Route. All test cases have passed. (Conducted by Nolan)
* IT401 – IT404 tests both assignPackage() and checkSpace() functions, which cover the requirements for Max Weight on Truck, Shipment Weight, Shipment Size, Getting the Shortest Path, Shipment Address, and Functional Truck Route. All test cases have passed. (Conducted by Nolan)

ACCEPTANCE TEST CASES RESULTS: (based on acceptance-test-cases-g3)

# Evaluation

This section provides an overall evaluation of the testing process including problems (not fixed issues) and limitations.

There were some bugs, with divert and assignPackage, this resulted in failed tests, and tests not running, these resulted in debugging which took some time, the errors lied in the test data as well as the function storing values/calculating values a wrong way. Besides this, the overall testing, blackbox, whitebox went very well, acceptance tests did as well.

# Summary of Activities

This section provides a summary of the major testing activities and events. This section also summarises testing resource information, such as total staffing levels, total testing time, cost, etc.

Summary Of Testing:  
  
Overall, the testing pointed out some bugs which needed to be addressed as well as logic changes in the custom implemented functions. The integration and acceptance tests allowed for the code to be changed, to meet the requirements. (as well as pointing out syntax errors, semantic errors, etc). Furthermore, the testing plan greatly aided in helping to form a proper plan, which included scheduling of when to implement data structures, custom functions, as well as when to start creating the necessary tests and when they should be finished. This allowed us to mostly stay on schedule since we had clear goals for each milestone, and ensured that all necessary documentation was updated with results by the end of the week. Lastly, the total testing time was about 3 weeks. This resulted in quality test data, and testing overall, which minimized bugs/errors.

Schedule of All Activities Done:

* Milestone 1: Created the Group Contract, GitHub repository, and Jira Project. Scrum report was also completed on time.
* Milestone 2: Conducted Data Analysis, requirements analysis, implemented data structures, and completed the Test Plan along with the Test Strategy. Scrum report was also completed on time. All necessary documents were uploaded and updated in GitHub and Jira.
* Milestone 3: Defined 4 custom functions in the header file (dataStruct.h), started implementing the defined custom functions (functionImplementation.c), created black box test cases in document, and implemented 1 black box test into the Testing project which was linked to the source code project. Also, we created the requirements traceability matrix, updated the Jira project, and completed the scrum report on time.
* Milestone 4: Finished black box tests implementation with test results, finished the custom function implementations, created white box tests in an excel file, implemented 1 white box test into the Testing project, updated the traceability matrix with all test results, and uploaded most members’ hook files in the Github repository. The Jira project was updated and the scrum report was completed on time.
* Milestone 5: Finished white box tests implementation with test results, created integration test cases in the excel file, implemented all integration tests with results, created acceptance test cases in the excel file, and updated the traceability matrix with all test results. All necessary files were uploaded to GitHub repository, updated the Jira project, and the scrum report was completed on time. However, implementation of acceptance tests was moved to the next milestone.
* Milestone 6: Finished acceptance tests implementation with results, completed the final testing report, updated the traceability matrix with all test results (passed and failed), updated Jira project, uploaded necessary files to GitHub, and the scrum report was completed on time.