Test Plan ZCC Group B

**Objective :**

The test plan for our SFT221 Project encompasses detailing the strategies, processes, objectives, and methodologies employed during the testing phase. Our primary objectives include ensuring that the software aligns with the functional system requirements mentioned in the project instructions, identifying, and resolving any potential issues or bugs, and effectively documenting and communicating the test results to the team. Through systematic testing, we aim to validate the functionality and reliability of the software, thereby enhancing its overall quality and performance.

**Scope:**

The test plan will cover the testing of all functions within the software. The plan will define the unit, integration, end to end, and regression testing plan. The plan will include testing of all system use case requirements outlined in the project instructions, as well as performance and security testing to ensure that the program is free of bugs.

**Test Strategy:**

These are the types of testing we will do in order to achieve a successful program without bugs.

**Blackbox Unit Testing**: Where we will divide the test into two categories: testing general use-case scenarios, and exploratory testing which tests the limits of the code. It may take roughly 1 hour to design the tests, and 30 mins to execute them.

**Whitebox Unit Testing**: Where we will analyse the code for defects to find more test cases. It may take approximately 1 hour total to design and execute the Whitebox tests.

**Integration Testing**: Make another set of test cases once the functions are integrated into the program and create general-use test cases as well as exploratory tests for the software. It should take around 2 to 2.5 hours to fully design and execute the integration tests.

**Acceptance Tests**: These test cases will be designed after the software-requirements outlined in the project’s pdf. It will take roughly 1 hour to prepare the tests, and 20 mins to execute them.

**Procedure:**

**Selection Criteria:**

1. Truck selection is based on the shortest available path to the destination.
2. Trucks cannot traverse through buildings.
3. If distances between 2 paths are equal, the truck with the lighter load is chosen.
4. Distance is calculated using the Pythagorean theorem.
5. Trucks cannot move backward during their route.
6. Destinations must fall within the predefined delivery range, specified by the map's row and column.
7. Each grid position is listed from (x, y) coordinates, such as A1.

**Box Specs:**

1. Small Box: 0.25 cubic meters
2. Medium Box: 0.50 cubic meters
3. Large Box: 1.0 cubic meters
4. All boxes are perfect cubes with uniform dimensions on all sides.

**Truck Specs:**

1. Maximum weight load: 1000kg
2. Maximum volume: 36 cubic meters

**Test Coverage:**

1. Examination of upper and lower bounds for each function.
2. Analysis of upper and lower bounds for each parameter type.
3. Detection of function misuse, including incorrect data type parameters and passing NULL.
4. Verification of display/printing accuracy with both correct and incorrect parameters.
5. Performance checks.
6. Detection of unused parameters, variables, and functions.
7. Validation of the accuracy of calculated distances to prevent miscalculations.

**Paths Constraint:**

1. All paths will be fixed unless a truck needs to make a detour to deliver a package outside of the path.
2. The path’s starting position is fixed at 1A.

**Test Data:**

The test data will be generated using following process:

* work on possible use-case scenarios.
* break down each scenario and determine what type of function is needed to implement the user action.
* Develop Blackbox test data based on function prototypes, and use-case scenarios.
* Develop Whitebox test data based on the implemented functions.

**Planned execution:**

MS3 - Document a set of Blackbox tests for the functions, Blackbox test code

MS4 - Execute the Blackbox tests, Create Whitebox tests based on implemented functions, Execute the Whitebox tests, Debug and re-execute tests.

MS5 - Create integration tests, create acceptance tests, Execute the integration tests, Debug and re-execute tests, Update test-matrix.

MS6 Execute acceptance tests, Debug and re-execute tests, Create the final test report.

**Members will be assigned tasks in each milestone on Jira.**

**Environment:**

**JIRA** – to be utilized to manage and keep track of our testing. Jira is a tool to organize projects, and for our test plan, we will use Jira to keep track of which software tests needs to be written, executed, and documented.

**GitHub** - For each milestone, we will commit any deliverables onto GitHub. This includes final test documentation, reports, and source codes for this project.