Test Plan - SFT221 ZCC Group C

1. **Introduction**

The project involves optimizing the routing for three delivery trucks based on various constraints, including cargo capacity and Euclidean distance calculations, avoiding obstacles, and determining the most efficient paths for delivery. The objective of this test plan is to ensure that the routing optimization software works as intended, verifying that the software correctly assigns packages to the appropriate truck based on capacity constraints and shortest path calculations. The goal is to identify and fix any issues before the next development cycle.

1. **Scope**
   1. **In Scope:**
      1. Testing the assignment of packages to trucks based on weight, volume, and shortest path.
      2. **Verifying that the trucks avoid obstacles (buildings) on their routes.**
      3. Ensuring correct handling of edge cases, such as invalid input values.
   2. **Out of Scope:**
      1. Integration with the actual delivery system**.**
      2. User interface testing (as it is assumed to be a backend service).
2. **Test Strategy**

**Test Types:**

* **System Test:** Verifying the overall functionality of the routing algorithm.
* **Performance Test:** Ensuring the software can handle many shipments without significant delays.
* **Security Test:** Ensuring that the software is secure from threats.
* **Automated Test:** Using automated testing tools to ensure the software runs correctly under various conditions.
* **Stress and Volume Test:** Testing the software under maximum load conditions to ensure stability.
* **Recovery Test:** Ensuring the software can recover from crashes or other issues.
* **Documentation Test:** Verifying that all necessary documentation is complete and accurate.
* **Beta Test:** Conducting tests with a limited user base to find any remaining issues.
* **User Acceptance Test:** Conducting tests to simulate real-world usage and validate that the software meets business requirements.

1. **Requirement Understanding:** Detailed analysis of the project requirements.
2. **Traceability Matrix:** Mapping test cases to specific requirements.
3. **Prepare Test Cases:** Creating test cases based on the requirements.
4. **Review:** Peer review of the test cases to ensure coverage and accuracy.
5. **Environment Requirements**

**Hardware:** Development workstations with the necessary software installed.

**Software:** Development environment set up with the latest version of the routing software. Testing tools and frameworks as required.

1. **Execution Strategy**

**Entry Criteria:**

Completion of the initial development phase.

Availability of the test environment and necessary tools.

**Exit Criteria:**

All test cases executed. 95% of test cases passed, with no critical defects remaining.

1. **Critical:** System crashes or produces incorrect results.
2. **High:** Major functionality issues that affect operations.
3. **Medium:** Issues that degrade the quality but have workarounds.
4. **Low:** Minor errors that do not significantly affect functionality.
5. **Cosmetic:** UI issues that do not affect functionality.

**Test Reporting:**

**Reporting Frequency:**

1. Daily reports summarizing the number of tests conducted, passed, and failed.
2. Detailed reports on critical and high-severity defects.

Regular meetings between the QA team, developers, and project managers to discuss progress and issues.

The quality assurance team will have regular interactions with the developers through scheduled meetings and collaborative platforms to discuss and resolve any defects found during testing.

1. **Test Schedule**

**Milestones:**

* **Milestone 1:** Initial setup and basic functionality tests.
* **Milestone 2:** Advanced functionality tests and edge cases.
* **Milestone 3:** Performance and stress tests.
* **Milestone 4:** User acceptance tests.

**Estimated Duration:** Each milestone is expected to take one week, with adjustments based on actual team-progress.

1. **Control Procedures**
   1. 6.1 Reviews: regular code and test case reviews.

6.2 Bug Review Meetings: Weekly meetings to review and prioritize defects.

6.3 Change Request: Documenting and tracking any changes to requirements or test cases.

6.4 Defect Reporting: sing the project's defect tracking system to log and manage defects.

1. **Functions To Be Tested**

**• Shipment assignment to trucks.**

**• Calculation of Euclidean distances.**

**• Pathfinding avoiding obstacles.**

**• Capacity constraints handling.**

1. **Resources and Responsibilities**

9.1. Resources

* QA team member. Development team support.

9.2. Responsibilities

* QA team responsible for test case creation and execution.
* Development team responsible for fixing defects.

1. **Deliverables**

* Test cases.
* Test reports.
* Defect logs.

1. **Suspension / Exit Criteria**

* **Testing can be suspended if critical environmental issues arise.**
* **Exit criteria as defined in the execution strategy.**

1. **Resumption Criteria:** Resumption upon resolution of suspension criteria.
2. **Dependencies**

12.1 Personnel Dependencies: Availability of QA and development team members.

12.2 Software Dependencies: Availability of the latest build of the routing software.

12.3 Hardware Dependencies: Test environment setup.

12.3 Test Data & Database: Accurate and relevant test data.

1. **Risks**

13.1. Schedule: Delays in development affecting testing timelines.

13.2. Technical: Potential issues with test environment setup.

13.3. Management: Changes in project scope affecting test plans.

13.4. Personnel: Unavailability of key team members.

13.5 Requirements: Incomplete or unclear requirements affecting test coverage.

1. **Tools: Jira, GitHub, MS TEAMS**
2. **Documentation:** 1) Test plan. 2)Test cases. 3)Test reports.
3. **Approvals:** Approval from project stakeholders upon completion of the test plan.