Test Plan Template

1. **Introduction -Han**The primary aim of this test plan is to validate the efficacy of our package distribution algorithm. This algorithm is designed to optimally allocate packages to delivery vehicles, considering factors such as vehicle capacity, proximity to delivery locations, and the most efficient routes. Our goal is to ensure high accuracy in distribution, optimal route selection, and adaptability in scenarios where direct delivery may not be feasible.
2. **Scope-Han**

Assignment of parcels based on the carrying capacity of delivery vehicles.

Selection of the vehicle nearest to the delivery location.

Route optimization to ensure the shortest possible delivery time.

Management strategies for scenarios where direct delivery is obstructed.

1. **Test Strategy-Hanbi**

Our testing plan aims to make sure the system works well and does what it's supposed to. We'll do this by following a step-by-step process to check how the system performs, how it works, and if users are happy with it.

**<Testing Steps>**

Step 1: Trying Different Things:

* We’ll start by trying out different parts of the system to find any big problems. We’ll test things like how packages are handled, their weight, and where they're delivered. We'll also check if the system behaves the way it's supposed to.
* This includes checking the system itself, how fast it works, and making sure all the documents match up with what the system does.

Step 2: Making Sure Everything Works:

* Next, we'll make sure all the different functions of the system work correctly. We'll use automated tests to save time and check many parts at once. We'll also test how well the system handles when lots of people are using it at the same time.
* Additionally, we'll let real users try the system to see if it does what they expect.

**<Tests might not require>**

* Checking Security: We won't do this because there's no specific need mentioned in the project documents.
* Checking if the System Recovers: We won't do this because the system doesn't use certain features needed for recovery.

**<Tests might be required>**

* Understanding What's Needed: We'll carefully read and understand what the project needs, including its main features and how it's supposed to work.
* Keeping Track of Tests: Each team member will write down what they've tested in a list. This helps us keep track of what's been checked and what needs fixing.
* Writing Down Test Steps: We'll create step-by-step guides for each problem we find. This helps us make sure we're testing everything properly.
* Checking Each Other's Work: We'll share what we've found and fixed with the team. This way, everyone can make sure nothing's been.

**Environment Requirements**

Hardware:

Mac OS: The test environment requires Mac computers with sufficient hardware specifications to run the software being tested.

hardware specifications to run the software being tested.

Software:

Mac OS: The testing will be conducted using the latest version of macOS (e.g., macOS Big Sur) with required dependencies and software tools installed.

Test Harness: A pre-existing set of testing tools will be used to conduct the tests. These tools should be installed and properly configured on the test machines.

<Testing Strategy Worksheet>

|  |  |
| --- | --- |
| **Project Name**  winter24-sft221-nff-6 | **Author**  QA:Hao Han Kao |
| **Computing Environment**  Operating System: macOS  Development Environment: Visual Studio Code  Compiler: GCC | **Software Type**  Desktop application compiled with GCC in Visual Studio Code. |
| **User Demographics** | **Assumptions** |
| **Purpose of Test**  Manual Testing for GitHub Repository Access | **Phases of Testing**  Setting the private for out github repo in order to not allowing others can access it.   Give the link to our group members who do not login in github and see whether they can access or not . |
| **Scope of Testing**  Our whole files | **Critical Success Factors** |
| **Testing Types**  Security testing | **Tester Profiles**  Project group members. |
| **Development/ Test Tools**  GitHub's built-in repository settings and audit log feature will be used for manual review.  A checklist of expected configurations and security practices will guide the review process. | |
| **Business / Operational Concerns**  If our repositories are able to be accessed by random people, that would be a mistake. | |
| **Risks**  **Business**  **Technical**  **Project**  Our project are going to be changed by the others | |
| **Other** | |

1. **Execution Strategy-Han**
   1. This is the section where you will describe how heavy chests are executed. You can describe what the entry and exit criteria for the tests are. For example, you might be able to exit a test if it passes 95% of test scripts. In another situation, you might want to pass 100% of the tests. Or perhaps you want to declare but a test is completed if there are no severe or critical defects.
   2. You can describe the severity of defects in this section and break them down into severity levels of:
      1. **critical** which cause the system to crash or produce anomalous results,
      2. **high** which causes lack of program functionality and might have a work around,
      3. **medium** which is a bug which D crates degrades the quality of a system but often has a work around to give the desired functionality.
      4. **Low** which might be an unclear error message or some other minor error that has minimum impact on functionality
      5. **Cosmetic** which is something that makes the user interface less than optimal but still perfectly functional.
   3. **Test Reporting**

**Reports Overview:** For our group project, we will create reports that detail the outcomes of our testing phases. These reports aim to provide a comprehensive view of our testing process, the results obtained, and any issues identified during the tests.

**Frequency of Reports:** Reports will be compiled and shared with the group at the end of each testing day. This schedule ensures that all group members are promptly informed about the progress and can contribute to resolving any identified issues.

**Recipients:** The primary audience for these reports includes all group members involved in the project. By keeping everyone informed, we facilitate a collaborative effort in addressing and resolving testing challenges.

**Communication Strategy:** Feedback on the testing outcomes and any bugs discovered will be shared within the group through a structured reporting format. This process includes the following steps:

Compilation of daily testing reports highlighting key findings and status.

Discussion of these findings in our group meetings to plan for necessary actions.

Utilization of email and our group chat platform for ongoing communication and updates on issue resolution.

1. **Test Schedule-Han**

1.Projected Duration for Testing: It is anticipated that the testing phase will span roughly two weeks.

2. Expected Conclusion: We aim to conclude all testing activities by the conclusion of the third week, counting from when testing initially commences.

1. **Control Procedures-Han**

6.1 Quality Assurance Checkpoints: Ongoing checkpoints will be established to monitor the progress and adherence of the testing phase to the predefined project goals and specifications. These checkpoints will engage all critical stakeholders, including the project team, QA personnel, and project leadership, to ensure comprehensive oversight. The objective is to facilitate constructive feedback, identify any discrepancies early on, and adjust strategies as needed to enhance the project's output quality.

6.2 Issue Resolution Discussions: Scheduled discussions will take place to address and rank any discovered issues or bugs. These sessions will foster collaboration between the QA and development teams to evaluate bug reports, assess their impact, and delegate fix responsibilities. Such discussions are vital for keeping open lines of communication, overseeing the resolution process efficiently, and ensuring that fixes are implemented in a timely manner.

6.3 Modification Requests: During testing, the need for system adjustments or enhancements might emerge, possibly due to feedback from stakeholders, recognition of potential improvements, or shifts in project directives. A structured process will manage these modification requests, encompassing the documentation of changes, impact analysis, stakeholder approval, and eventual integration. This procedure is designed to ensure modifications are methodically vetted and integrated with minimal disruption.

6.4 Reporting Bugs and Issues: An integral component of the testing phase is the systematic documentation and reporting of any encountered bugs or issues. This will involve a uniform reporting template detailing the bug's nature, replication steps, expected versus actual outcomes, and any pertinent supporting information. Bugs will be classified by their severity level (such as critical, high, or low) and slated for prioritization and resolution. This structured reporting mechanism aims to streamline issue tracking, enhance communication among team members, and facilitate the swift resolution of problems.

1. **Functions To Be Tested-Han**

Core Functionalities Subject to Testing

7.1 Package Distribution Mechanism:

This critical functionality will undergo rigorous testing to confirm that it accurately distributes packages among delivery vehicles, considering parameters such as package weight, dimensions, and destination. We will simulate a variety of situations, including both typical and extreme cases, to assess the system's handling of different package volumes and vehicle capacities.

7.2 Optimal Route Determination Feature:

We will evaluate this feature to ascertain its capability in identifying the most efficient route for delivery between two points, factoring in the need to circumvent obstructions like buildings. The evaluation will encompass diverse origin-destination pairs and potential physical barriers to ensure robust route optimization.

7.3 Vehicle Load Capacity Assessment Tool:

Testing will be conducted on this tool to verify its precision in calculating the load capacity available in each delivery vehicle. Our scenarios will include a range of weights and package sizes to test the system's ability to accurately gauge and maximize vehicle utilization without exceeding load limits.

7.4 Delivery Information Compilation Function:

This function's accuracy in assembling and conveying detailed delivery information, including vehicle assignment, route diversions, and other pertinent details, will be meticulously tested. We will explore various delivery scenarios to ensure the generated messages are both precise and user-friendly, thereby facilitating smooth delivery operations.

1. **Resources and Responsibilities-Han**  
   8.1. Resources: For the upcoming testing phase, we will mobilize the following key resources:

a. Quality Assurance Team: A dedicated group of QA professionals will conduct the tests, meticulously record outcomes, and flag any issues or bugs encountered.

b. Testing Infrastructure: A robust testing environment will be set up, incorporating all necessary technological and simulated components to facilitate comprehensive testing.

8.2. Responsibilities:

a. QA Team: Members of the QA team are tasked with the thorough execution of all test scenarios, diligent documentation of findings, and prompt reporting of any bugs or issues.

b. Software Developers: The developer team will take charge of rectifying bugs reported by the QA team, applying code corrections, and performing subsequent retests to ensure issues are resolved.

c. Project Supervisors: Project leaders will coordinate the testing efforts, bridging the QA and development teams and ensuring the provision of resources needed for effective testing.

Adhering to these procedural guidelines and clearly defining team roles will help us assure the integrity, performance, and regulatory compliance of our package management system.

1. **Deliverables-Han**Accurate package assignment to vehicles, considering spatial constraints, proximity to delivery points, and detour necessities.

Precise computation of the most direct route from the vehicle’s current path to the delivery location, avoiding any obstructions.

Capability to manage situations where direct access to a delivery point is obstructed.

Generation of detailed notifications about chosen vehicles, delivery endpoints, and alternate paths when necessary.

1. **Suspension / Exit Criteria-Han**Inability of the algorithm to allocate packages correctly as per specified criteria.

Failure in accurately calculating or finding routes due to algorithmic errors.

Incapacity to address scenarios with inaccessible delivery locations.

Incorrect or incomplete information output by the algorithm.

Additional trigger for pause:

5. Cease operation if any package exceeds the weight limit of 1000 kg.

1. **Resumption Criteria-Han**Following the resolution of any flagged concerns or defects, the system will undergo another round of comprehensive testing. This is to verify that all project requirements are fully satisfied.

To assure the system's robustness and precision, it will be subjected to a variety of testing scenarios and cases anew. This step is critical for confirming the system's operational effectiveness across diverse conditions.

After any adjustments are made to the system, it's crucial to conduct additional tests. These tests are to ensure that recent changes have not inadvertently introduced any new problems or adversely affected the system's performance.

1. **Dependencies-Han**  
   12.1 Team Composition: Detailing the project team members, their specific roles, and responsibilities, while noting the project's reliance on certain key personnel’s expertise and availability.

12.2 Software Requirements: Identifying the critical software components required for testing, including the application under test, testing management tools, defect tracking systems, and necessary automation tools. Compatibility with the testing tools is essential for accurate and efficient testing outcomes.

12.3 Hardware Needs: Outlining the hardware requirements necessary for the project, including specific devices or components crucial for testing.

12.4 Data and Database Access: Access to an appropriate testing database or environment is crucial, especially for testing that involves data handling and database functionalities. Comprehensive test data covering a broad spectrum of cases is essential for thorough testing.

1. **Risks-Han**  
   13.1 Timing Constraints: Risks associated with meeting the testing schedule, including unforeseen delays or scope expansions.

13.2 Technical Challenges: Potential technical barriers, such as issues with infrastructure, test environment setup, or tool limitations.

13.3 Project Oversight: Risks linked to project management, including coordination difficulties or resource shortages.

13.4 Team Dynamics: Concerns related to the project team, like skill mismatches or communication gaps.

13.5 Specification Clarity: The risk of ambiguities or changes in project specifications impacting the scope or deliverables.

1. **Tools-Han**

Unit Testing Framework: Google Test

1. **Documentation-Han**

Emphasizing the importance of selecting appropriate testing tools to enhance efficiency and the creation of various documents (test plans, bug reports, etc.) to ensure clear communication and documentation throughout the testing process.

1. **Approvals-Han**

Defining the approval process for the test plan and related documents, including identifying who has authority for approvals and the timeline for securing these approvals, to ensure all necessary reviews are completed before testing begins.