TEST PLAN

For

Trawingo

Ver no. 0.1

Disclaimer: This document is classified as 'Internal' and used by Dalvkot Utility Enterprises Pvt. Ltd. employees only until and unless it is specified otherwise.

Document Revision History

Ver. No.	Date (dd-Mmm-yyyy)	Name of the Author	Change Information
0.1	24-Jan-2025	Sankitha P R	Initial draft version

Ver. No.	Date (dd-Mmm-yyyy)	Name of the Reviewer	Name of the Approver
0.1			

TABLE OF CONTENTS

1. INTRODUCTION 4

1.1 OBJECTIVE 4
1.2 SCOPE WRT UNIT/INTEGRATION/SYSTEM TESTING 5
1.3 DEFINITIONS AND ACRONYMS 7
1.4 ASSUMPTIONS, CONSTRAINTS, DEPENDENCIES, RISKS 8
1.5 DELIVERABLES 10
1.6 TEST TEAM 10

2. TEST STRATEGY 11

2.1 FEATURES TO BE TESTED IN UNIT/INTEGRATION/SYSTEM 11
2.2 TEST APPROACH 12
2.3 TESTING ENVIRONMENT SET-UP 12
2.3.1 Hardware 12
2.3.2 Software / Tools 13

3. TEST CRITERIA 13

3.1 ENTRY CRITERIA 13
3.2 SUSPENSION AND RESUMPTION CRITERIA 14
3.3 EXIT CRITERIA 15
3.4 ACCEPTANCE CRITERIA 15
3.5 BUG CLASSIFICATION STRATEGY 16

1. Introduction

- The objective of this test plan is to ensure the Trawingo web application meets the highest standards of quality and reliability through comprehensive manual testing. Given the critical nature of the application, maintaining superior quality is paramount. This document outlines the approaches and methodologies that will be employed for unit, integration,system testing,and regression testing and retesting of the Trawingo application.

- This test plan serves as a communication tool among team members, detailing the objectives, test responsibilities, entry and exit criteria, scope, schedule, major milestones, and overall approach. It clearly identifies the test deliverable and specifies what is considered in and out of scope.

- By adhering to this test plan, we aim to systematically and thoroughly verify that all aspects of the Trawingo application function as intended, ensuring a seamless and satisfactory user experience.

1.1 Objective

- **Automate and Optimize Transport Operations:** Streamline the end-to-end employee transportation process, reducing manual intervention and ensuring efficient scheduling and management.

- **Comprehensive Dashboard:** Provide a centralized dashboard to track vehicle movements,manage approvals, and monitor key performance metrics for informed decision-making.

- **Efficient Routing and Vendor Management:** Enable seamless routing, vehicle allocation,and vendor management to ensure smooth operations and optimal use of resources.

- **Enhance Operational Efficiency:** Leverage real-time tracking and reporting toolsto monitortrips, identify bottlenecks, and improve overall operational performance.

- **Ensure Safety and Compliance:** Maintain compliance with safety standards by integrating features such as driver document validation, SOS alerts, and emergency response mechanisms.

1.2 Scope wrt Unit/Integration/System Testing

a) Unit Testing Scope

Features to Be Tested

- Web Application Dashboard
  - Vendor Management: Standalone operations for adding, editing, and managing vendors.
  - Super Admin (Trawingo & Corporate): Core modules such as user management, trip management, route optimization, and policy enforcement.
  - Sub-Super Admin: HR, compliance, billing, and routing operations.
- Mobile Applications:
  - Driver App:
    - Individual screen functionalities (e.g., Trip Assignment, Live Navigation, SOS).
    - Navigation logic validation.
  - Employee App:
    - Trip Booking, Live Tracking, and Notifications components.
    - Emergency/SOS features.
- Back-end Components:
  - Validation of algorithms and logic for:
    - Route optimization.
    - Vehicle and driver assignments.
    - Policy enforcement mechanisms.

Features Not to Be Tested

- End-to-end workflows involving multiple components.
- Aesthetic/UI aspects not related to core functional behavior.

b) Integration Testing Scope

Features to Be Tested

- Web Application Dashboards:
  - Interactions between:

        - Super Admin and Vendor Admin for trip management.
        - Sub-Super Admin roles with system-wide compliance checks.
        - Corporate Admins and Vendors for coordination and notifications.
    - Data flow consistency across dashboards (e.g., trip and billing data synchronization).
- Mobile Applications:
    - Driver App:
        - Communication between the back-end and mobile app for trip assignments and navigation.
        - Synchronization of trip history, alerts, and notifications.
    - Employee App:
        - Integration of trip booking with live tracking and backend trip details management.
        - Alerts and emergency features synchronization.
- System Interfaces:
    - Integration between mobile apps (Driver and Employee) and the backend system for real-time data updates.
    - Third-party APIs for navigation and location tracking (basic checks).

Features Not to Be Tested

- Experimental or future features, such as advanced AI recommendations.
- Integration with legacy systems scheduled for deprecation.

c) System Testing Scope

Features to Be Tested

- End-to-End Functional Workflows:
    - Web Application:
        - Trip lifecycle across all dashboards (e.g., booking, assignment, live tracking, completion).
        - User management and role-based access validations.
    - Mobile Applications:
        - Complete trip booking and execution workflows for employees and drivers.

- Emergency and SOS processes, including notifications to admins.

- Cross-Platform Testing:
  - Synchronization between web and mobile platforms.
  - Real-time updates for notifications and live tracking.
  - Validation of role-based access control across platforms.

- Non-Functional Aspects:
  - Basic performance tests (e.g., responsiveness during peak traffic).
  - Usability tests for mobile app workflows.
  - Basic security checks for authentication and sensitive operations.

Features Not to Be Tested

- User manuals, training materials, and documentation.
- Stress testing and advanced performance bench-marking.
- Penetration testing for vulnerabilities (handled in a separate phase).

1.3 Definitions and Acronyms

a) Definitions

Term	Definition
Test Case	A set of conditions or variables used to determine if a software application behaves as expected.
Bug/Defect	An error, flaw, or fault in a software program that causes it to produce incorrect or unintended results.
Test Scenario	A high-level description of functionality to be tested, focusing on end-to-end workflows.
Test Script	Detailed step-by-step instructions for executing a test case.
Regression Testing	Testing conducted to verify that new changes have not adversely affected existing functionality.
Functional Testing	Testing focused on ensuring the system performs its intended functions correctly.
Non-Functional Testing	Testing focused on system attributes such as performance, scalability, and usability.
Acceptance	Specific conditions or outcomes that must be met for a feature or

Criteria	release to be accepted.
Severity	The impact level of a defect on the system or application.
Priority	The order in which a defect should be fixed based on its importance and impact on the system.

b) Acronyms

Acronym	Full Form
UAT	User Acceptance Testing
SIT	System Integration Testing
RTM	Requirements Traceability Matrix
UI	User Interface
API	Application Programming Interface
P1, P2, P3, P4	Priority Levels (e.g., P1 = High Priority, P4 = Low Priority)
SDLC	Software Development Life Cycle
STLC	Software Testing Life Cycle

1.4 Assumptions, Constraints, Dependencies, Risks

1.4.1 Assumptions

- All required test environments (e.g., staging, QA, and production-like environments) will be available and fully configured before testing begins.
- Functional and non-functional requirements are complete, signed off, and do not change significantly during the testing phase.
- All necessary tools (e.g., automation frameworks, bug-tracking systems, and test management tools) are accessible and functional.
- Test data will be accurate, complete, and readily available at the start of test execution.
- Sufficient time will be allocated for defect resolution and re-testing activities.

- The development team will provide timely support for fixing defects and clarifying requirements.

1.4.2 Constraints

- Limited budget restricting the acquisition of advanced testing tools or external resources.
- Testing timelines are compressed due to tight project schedules.
- Hardware or software limitations, such as server availability or outdated infrastructure.
- Limited access to third-party systems or APIs for integration testing.
- Restrictions on testing scope based on client or business priorities.
- Dependency on external teams for environment setup or data provisioning.

1.4.3 Dependencies

- Timely delivery of build versions from the development team for testing.
- Availability of functional and stable environments for executing various test phases.
- Collaboration with dependent teams (e.g., DevOps, API integration teams) to align on timelines.
- Access to real-world production data or representative test data.
- Support from third-party vendors for integrated systems or tools.
- Completion of prior testing phases (e.g., unit testing or integration testing) before initiating system or acceptance testing.

1.4.4 Risks

- Delays in environment setup or unavailability of critical test data.
- Late discovery of high-priority defects, leading to rework and timeline overruns.
- Key resources becoming unavailable during critical phases of testing.
- Scope creep or late requirement changes impacting test coverage.
- Poorly documented requirements leading to ambiguous or missing test cases.
- Performance issues with the application during load or stress testing.
- Dependence on external teams or vendors causing potential delays in integration testing.

1.5 Deliverables

Sl. No.	*Name*	*Delivery Date*
1.	Test Plan Document	29-Jan-2025
2.	Test Scenario and Test Cases Document	dd-Mmm-yyyy
3.	Defect Tracker Report	dd-Mmm-yyyy
4.	Test Execution Report	dd-Mmm-yyyy
5.	Traceability Matrix	dd-Mmm-yyyy

1.6 Test Team

Role	**Staff Member**	**Responsibilities**
Project Owner	Durga	1.Acts as a primary contact for development and QA team. 2.Responsible for Project schedule and the overall success of the project.
QA Lead	Sudarshan J Ramya	1. Participation in the project plan creation/update process. 2. Planning and organization of test process for the release. 3.Coordinate with QA analysts/engineers on any issues/problems encountered during testing. 4.Report progress on work assignments to the Project Owner
QA	Sankitha Lavanya Krishna	1.Understand requirements 2.Writing and executing Test cases 3.Preparing Requirement traceability matrix. 4.Reviewing Test cases, RTM 5.Defect reporting and tracking 6.Retesting and regression testing

		7.Bug Review meeting 8.Preparation of Test Data Coordinate with QA Lead for any issues or problems encountered during test preparation/execution/defect handling.

2. Test strategy

- It defines the scope, risks, quality criteria, testing techniques, resources, and timelines, providing a strategic view of how testing will be conducted throughout the project.

2.1 Features to be tested in Unit/Integration/System

Sl No.	*Features to be Tested*	*Features not to be Tested*
1.	**Functional Requirements:** User authentication (login, logout, password reset)	**Training Materials:** User manuals, help guides, and other documentation materials.
2.	**System Interface:** Integration with payment gateways and notifications systems	**System Interfaces:** Functionalities provided by third-party travel information providers and map services (external).
3.	**Infrastructure Components:** Mobile app performance on supported devices and operating systems	**Infrastructure Components:** Extensive security penetration testing and in-depth performance bench-marking beyond load tests.
4.	**Web Dashboards:** Vendor Management, Super Admin (Corporate/Organization), Sub-Super Admin	**Future Features:** Advanced itinerary management, customization features, and new payment methods under development.
5.	**Mobile App (Driver):** Login/Signup, Trip Assignment, Navigation, Passenger Check-In, and Notifications	**UI/UX Changes:** Minor cosmetic changes that do not affect core functionality.
6.	**Mobile App (Employee):** Trip Booking, Live Tracking, Emergency/SOS, Profile Settings, Feedback and Ratings	**Experimental Features:** AI-driven travel recommendations not ready for release.
7.	**Process Requirements**: Compliance enforcement (e.g., policy adherence)	**Internal Tools:** Administrative tools for the support team covered in a separate testing cycle.
8.	**Data Management:** Reporting and analytics for Vendor Admin	**Process Requirements:** Integration with older travel booking systems being phased out.
9.	**Communication and Notifications:** Issue resolution, trip updates, and	-

	escalations in Super Admin dashboards	

2.2 Test Approach

Sl. No.	*Testing Type*	*Artifacts*	*Responsibility*
1.	Acceptance	Acceptance Test case Acceptance test Data	Project Manager QC Customer
2.	System	System Test Cases System Test Data	Tester
3.	Integration	Integration Test Cases Integration Test Data	Tester
4.	Unit	Unit Test Cases Unit Test Data	Developer

2.3 Testing Environment Set-up

It describes the environment needed to validate the product or its components. The environment may be purchased or set up internally. To improve cost efficiency and productivity, integration and verification environments may be combined.

2.3.1 Hardware

Sl. No.	Item	Description	No.
1	Test Servers	These are actual hardware servers with dedicated resources (CPU, RAM, disk space).	
2	Workstations	Workstations for manual and automated testing(e.g., high-performance PCs with Intel Core i5, 16GB RAM)	
3	Mobile Devices	Mobile devices for mobile app testing (e.g., iPhone, Samsung Galaxy S21 for testing mobile applications on different platforms).	

2.3.2 Software / Tools

Sl. No.	Item	Description	No.
1	Test Management Tool	JIRA, TestRail for test case management,Microsoft excel,WPS	
2	CI/CD Tools	Jenkins, GitLab CI for continuous integration and deployment	
3	Database	MYSQL for database testing	
4	OS and Browsers	Windows,Chrome, Edge for cross-platform testing	

3. Test Criteria

3.1 Entry Criteria

i. **Availability of Test Items:**
All the components or features to be tested (e.g., software builds, versions, or modules) are available and have been delivered in a stable state, meeting the agreed-upon quality criteria.

ii. **Availability of Test Plan:**
The test plan document has been reviewed, approved, and distributed to all relevant stakeholders. This includes the test objectives, scope, approach, resources, and schedule.

iii. **Availability of Test Resources:**
Necessary resources (including hardware, software, test environments, and human resources) are available and ready. This also includes ensuring that test environments are set up, required tools are available, and all team members are assigned to their roles.

iv. **Test Data Availability:**
Test data, including both valid and invalid inputs, is prepared and available for use. Any specific configuration or data preparation required for testing has been completed.

v. **Completion of Requirements and Design Documents**:
All functional and non-functional requirements are clearly defined, and test cases can be mapped to them. Design documents, including architecture or design specifications, should be available for reference during testing.

vi. **Previous Defects Resolved**:
Any high-priority defects identified in previous phases (e.g., development or unit testing) must be addressed and closed before the testing process starts.

vii. **Test Environment Readiness**:
The test environment should be configured, accessible, and aligned with the production

environment where necessary. The test environment should replicate the production setup and be stable for testing purposes.

3.2 Suspension and Resumption Criteria

3.2.1 Suspension Criteria:

Testing will be suspended if any of the following occur:

- **Critical Defects**: If the number or severity of critical defects exceeds a threshold where follow-on testing is ineffective, or if defects are found that halt the functionality of key features. For example, a defect that causes the application to crash or leads to a data loss.
- **Test Environment Issues:** If the test environment becomes unavailable, unstable, or significantly differs from the expected configuration. This could include issues such as unavailability of hardware or key services.
- **Resource Unavailability:** If key testing resources (e.g., test personnel, hardware, or software tools) become unavailable for an extended period, testing may be suspended until the issue is resolved.
- **Incomplete Test Data:** If the necessary test data is not available or is incomplete, halting testing until the data is adequately prepared.
- **Blocked by External Dependencies:** If testing is dependent on an external system or team, and their deliverables or resolutions are delayed, testing will be paused until these dependencies are met.

3.2.2 Resumption Criteria:

Testing will resume when:

- **Defects are Resolved:** Critical defects or blocking issues have been addressed, verified, and resolved. The test team will ensure that resolved defects have been retested before proceeding.
- **Environment Restored:** The test environment issues are resolved, and the environment has been restored to the agreed-upon configuration.
- **Resources Become Available:** Once key personnel, hardware, or other resources are available, testing will continue.
- **Test Data is Complete:** If test data was incomplete, testing will resume once the missing data is made available.
- **External Dependencies Met:** Once external dependencies or third-party systems are available or resolved, testing will proceed.

3.3 Exit Criteria

i. **Test Case Execution Completion**

a) All planned test cases (functional, integration, system, performance, etc.) have been executed.

b) A minimum percentage (e.g., 95-100%) of test cases must pass.

ii. **Defect Resolution**

a) All critical and high-priority defects are resolved and verified.

b) Medium and low-priority defects have been documented, acknowledged, or deferred with proper justification.

c) No blocker or showstopper defects remain open.

iii. **Test Coverage Requirements Met**

a) The defined test coverage criteria (e.g., requirement coverage, code coverage, branch coverage) have been achieved.

b) All business-critical functionalities have been tested.

iv. **Regression Testing Completion**

a) All regression test cases have been executed successfully after defect fixes.

b) No new critical defects introduced by recent changes.

v. User Acceptance Testing (UAT) Sign-off

a) UAT has been successfully conducted, and stakeholders have provided approval.

3.4 Acceptance Criteria

i. General Acceptance Criteria

The test results will be accepted if:

a) **Functional Requirements**

- All **critical and core functionalities** are working as specified in the requirements document.
- No **major issues** or deviations from expected behavior are found.
- All essential business workflows (such as login, payment, user management) operate correctly.

b) **Defects Resolution**

- **All critical and blocker defects** have been resolved.
- **High-severity defects** are either fixed or deferred with stakeholder approval.
- **Minor defects** that do not impact the business process have been documented.

c) **Performance & Security Compliance**

- The software meets or exceeds defined **performance criteria** (e.g., load time, system responsiveness).
- No **high-severity security vulnerabilities** exist, and the system has passed essential security tests.

d) **User Acceptance Testing (UAT) Sign-Off**

- All **UAT test cases** have been executed, and the system has passed acceptance by the business stakeholders.
- Feedback from **end-users** has been incorporated, and any necessary changes have been made.

e) **Regression Testing**

- Regression testing has been completed and ensures no **existing functionalities** have been impacted by new changes.
- The system behaves **consistently** across all supported environments (Test, Staging, Production-like).

f) **Test Deliverables Completion**

- **Test summary reports, defect reports, and all testing artifacts** have been completed and reviewed.
- **Compliance and audit checks** have been passed for the testing phase.

3.5 Bug Classification Strategy

- The Bug Classification Strategy defines how defects or bugs identified during testing will be categorized. This ensures that bugs are prioritized appropriately, allowing the team to focus on critical issues first and effectively manage the testing process. Below are the categories for classifying defects:

i. Severity Levels

Defects will be classified based on their **impact** on the system and its functionality:

Severity Level	Description	Action
Critical	● A defect that causes the application to crash, freeze, or become unusable. ● Major loss of functionality that prevents users from proceeding with essential tasks (e.g., login failure). ● Security vulnerabilities that may expose sensitive data.	● Fix immediately. ● Regression testing after the fix.
Major	● A defect that severely impacts functionality, but does not cause a system crash. ● Can lead to incorrect results, significant system behavior deviation, or data corruption. ● Functionality is impaired but workarounds may exist.	● Fix before release. ● Priority for retesting and regression testing.
Minor	● A defect that has minimal impact on functionality or user experience. ● Examples: typos, slight UI misalignment, or minor inconsistencies in non-critical features.	● Can be fixed in future releases. ● No immediate impact on delivery.

ii. Prioritization

Based on severity, impact, and business requirements, defects will be prioritized as follows:

Priority Level	Description	Action
P1 (Highest)	Critical defects that need to be addressed before release or immediately after discovery.	Fix in the current cycle.
P2 (Medium)	Major defects that impact usability but do not prevent use of the system.	Fix during the current testing cycle or next minor release.
P3 (Low)	Minor issues that do not affect overall functionality or performance.	Address in future updates.

iii. Defect Types

Defects can also be classified based on the type of issue they represent:

Defect Type	Description	Action
Functional	Defects in the core functionality of the application, such as incorrect calculations, errors in business logic, or failure of critical workflows (e.g., payment processing, login).	Immediate fix required, depending on severity.
Structural	Defects related to the architecture of the system or its components, such as issues with database design, server configurations, or integration points.	Requires thorough investigation and may require architectural adjustments.
UI/UX	Defects related to user interface or user experience, including issues like misalignment, broken links, inconsistent icons, or poor design practices.	Fix when the defect impacts the user experience or usability.
Performance	Defects related to the performance of the system, such as slow page load times, delays in processing, or high resource consumption under load.	Must be resolved before release, especially if it affects system scalability or user experience.
Security	Defects related to security vulnerabilities in the system, such as unauthorized access, data leaks, or authentication failures.	High priority; must be addressed immediately.
Compatibility	Defects caused by issues with browser compatibility, OS compatibility, or interactions with third-party tools or services.	Should be resolved for all supported platforms and browsers.
Integration	Defects occurring when external systems or modules fail to interact as expected (e.g., APIs, third-party services).	Fix required if it impedes the integration flow or critical business operations.

Template Revision History (For SEPG only)

Ver. No.	Date (dd-Mmm-yyyy)	Name of Author	Change Information
0.1	8/Apr/2024	Chetan Babu.C	Initial Draft Copy
1.0	12/Apr/2024	Chetan Babu.C	Baselined QMS v1.0
1.1	23/Oct/2024	Chetan Babu.C	Updated the QMS
2.0	4/Nov/2024	Chetan Babu.C	Baselined QMS v2.0

Ver. No.	Date (dd-Mmm-yyyy)	Name of the Reviewer	Name of the Approver
0.1	12/Apr/2024	Vishnu Varadan	Vishnu Varadan
1.1	4/Nov/2024	Vishnu Varadan	Vishnu Varadan

Effective Date (Release date of the QMS document)	**11/Nov/2024**