TEST PLAN

SOFTWARE QA TESTER PROJECT TEST

|  |  |
| --- | --- |
| General Information | |
| Customer | Commsult Project Test |
| Created by | Sultan Ahmad Alfath |
| Preparation date | 17-Oct-24 |
| Version | V.0.1 |
| Status | Draft - ongoing |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Revision History | | | | | |
| Version | Description | Author | Date | Approved by | |
| Author | Date |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Contents

[1. Introduction 4](#_Toc180017520)

[2. Test Items 4](#_Toc180017521)

[3. Feature to be tested 5](#_Toc180017522)

[4. Features Not to be Tested 6](#_Toc180017523)

[5. Test Approach 6](#_Toc180017524)

[6. Test Criteria 8](#_Toc180017525)

[7. Test Deliverables 8](#_Toc180017526)

[8. Testing Schedule 9](#_Toc180017527)

[9. Roles and Responsibilities 9](#_Toc180017528)

[10. Risks and Contingencies 9](#_Toc180017529)

[11. Test Management 9](#_Toc180017530)

[12. Approval and Sign-off 15](#_Toc180017531)

# 1. Introduction

1.1 Purpose

* Creating test case to identify defect on system with new-feature before release
* Defining list of test deliverables
* Identifying software components to be tested

1.2 Scope

1.2.1 Scope of driver part

The following components of mobile software functionality will be tested:

1. Drag-and-drop feature
2. Order list feature
3. Update order list feature
4. Confirmation of delivery stops after changing/updating order list

1.2.1 Scope of driver part

The following components of Warehouse Management System software will be tested:

1. Warehouse notification feature

1.3 Objectives

The testing objective is to ensure that the new feature does not create defects in existing features and that it functions precisely as defined in the requirements document.

1.4 References

Test plans are using following references:

1. Software QA Tester Project Test (product requirements document)
2. Use cases of driver and warehouse analyst
3. Testrail
4. Regression testing suites

# 2. Test Items

Following items will be tested based on existing version of software:

1. Real-time notification system
2. Delivery order distribution
3. Mobile App: delivery process
4. Legacy feature for regression testing

# 3. Feature to be tested

Following test items will be tested based on PRD, with additional regression test to ensure existing feature functionality, performance test, and usability testing for user experience.

* + **Feature functionality specific test (Based on PRD):**

1. **Update delivery order list by driver via mobile app**  
   Test verifies that driver can update order list via mobile app and see updated order list without waiting for confirmation or delay.
2. **Notification received by warehouse management system from driver mobile app**  
   Test verifies that warehouse management system can receive notification of updated list.
3. **Delivery confirmation on updated order list**  
   Test verifies that driver can send an accurate confirmation data to warehouse management system after updating order list via mobile app.
4. **End-to-end test with update order list feature**Test verifies that driver can execute usual business flow by using new drag-and-drop feature to update order list.
   * **Regression test:**
5. **Distribution of order list created by analyst via back end to driver mobile app**  
   Ensuring the analyst could send order list to driver app via back-end system.
6. **Notification received by warehouse management system from driver mobile app**Ensuring warehouse management system receive notification from driver app when driver update the order list and confirming completed delivery.
7. **List of original delivery order inside driver mobile app**Ensuring driver can see original order list sent by analyst.
8. **Delivery confirmation on updated order list**Ensuring driver can confirm completed delivery on updated order list.
9. **Delivery confirmation on original order list from analyst**Ensuring the driver can confirm completed delivery on original order list from analyst
10. **End-to-end test without update order list feature**Ensuring the system and both user (analyst and driver) can do usual business flow without using new drag-and-drop feature to update order list.
11. **End-to-end test with update order list feature**Ensuring the system and both user (analyst and driver) can do usual business flow with using new drag-and-drop feature to update order list.
    * **Usability test:**
12. **Drag-and-drop feature ease of use on driver mobile app**Evaluates how intuitive and user-friendly the feature on mobile app for various driver age (assumption that age influence the tech savviness)
13. **UI feedback for updating the order list and error handling**Evaluates how communicative the UI when giving feedback to driver after updating the order list and handling error (e.g signal lost after drag-and-drop)
14. **Error recovery**Evaluates how effective the UI and system to be recovered from user mistakes (e.g accidental drag-and-drop)

# 4. Features Not to be Tested

Some legacy features will not be tested by user and manual QA engineer, such as:

1. Login
2. Create order by analyst
3. Sorting stops for order list by analyst
4. Back-end related test (no details provided in the requirement document)

These features will not interfere with the new feature, however these features will consist inside regression testing in end-to-end test. Back end related test also not going to be executed since the requirement document didn’t specifies any of back-end systems, so we are working on assumption that the back-end working well or covered by other team.

# 5. Test Approach

5.1 Testing Levels

* + **Unit testing**. This level focuses on individual components of code with isolated test environment (only for the function or code) and usually performed by developers.
  + **API testing.** This level focuses on API to ensure it returns the correct response for various input and handle error conditions appropriately
  + **Integration testing**. This level focuses on interaction between integrated system/components of software to validate data exchanges between system. Mostly back-end and front end or back-end and database.
  + **User Acceptance Testing (UAT)**. Conducted by end user to validate software against requirement document/business requirements to ensure it meets the needs of user as a last step of testing before deployment.

5.2 Testing Types

* + **Functional Testing.** Ensuring the software works as expected and aligned with product requirement document.
  + **Usability Testing**. Assessing user experience on UI design of new-feature. How intuitive, and user-friendly the design is.

5.3 Test Environment

Environment used in testing:

* + **Staging**
  + **Sandbox (environment with production server duplicate data)**
  + **Production**

Tools used in testing:

* + **Postman**
  + **Swagger**
  + **Android/iOS emulator**
  + **JIRA**
  + **Selenium/Katalon/Robot framework (regression testing)**
  + **Maze.co (usability test/UX)**

5.4 Test Methodologies

**Manual testing** will be conducted on following test:

* + **Functional testing**. Involves WMS notification system, driver drag-and-drop feature, and delivery confirmation feature by using gherkin language for test cases creation
  + **Usability testing.** Involves drag-and-drop ease of use, how user interact with new feature without tutorial, and how the user satisfied with UI feedback when facing error.
  + **UAT testing.** Ensuring all requirements are met and user satisfied with the result

**Automated testing** will be conducted on following test types:

* + **Functional regression test**. Involves functional regression testing with legacy and new-feature before release, with end-to-end test.

# 6. Test Criteria

6.1 Functional Test Pass Criteria

* + Driver can update order list after receiving order list from analyst
  + Driver can confirm updated order list
  + Driver can’t update order list if the card were dropped outside the order list zone
  + Driver can’t update completed order card position
  + Driver can’t update if the card were dropped in the same position
  + Analyst can receive and view updated order list notification in WMS from backend system
  + Analyst can receive and view order confirmation notification in WMS from backend system with real-time timestamp and total completed deliveries.
  + API system can receive and store request data
  + Backend system successfully send webhooks to WMS API

6.2 Usability Test Pass criteria

* + 80% user or more, can complete tasks without tutorial
  + Below 5% of users are encountered error to do given task
  + 80% user or more, can recover from error without assistance
  + 80% user or more agree they have clear feedback and satisfied with UI design

6.3 UAT Test Pass criteria

* + All requirements are met by system
  + User satisfied with result

6.4 Exit Criteria

The exit criteria for this test plan is when all test criteria have been passed.

# 7. Test Deliverables

Following deliverables will be available at the end of test:

* Test plan
* Test cases
* Test scripts
* Defect documentation
* Test report
* UAT result

# 8. Testing Schedule

Dummy schedule.

# 9. Roles and Responsibilities

Sultan with full responsibilities to test all feature

# 10. Risks and Contingencies

There are following risks with their mitigation:

1. Incomplete requirement. Engage stakeholders in development process to clarify the problem
2. Resource Constraint. Planning resource allocation and discussing with team for alternatives.
3. Changing requirement. Updating test plan, test cases, and all test related document. Re-test all test cases.
4. Failed user adaptation of new feature. Creating training or tutorial for new feature.

# 11. Test Management

11.1 Test Case Design

Test case design will be made using gherkin language and excel for documentation. All of test cases will be imported to test rail for archive. The header of excel consist test-id, test title, pre-condition, steps, expected result, actual result, test priority, and tester name. There are 3 test cases, with following cases.

11.1.1 API test case (for notification)

| Test-ID | Test Title | Pre-condition | Steps | Expected result |
| --- | --- | --- | --- | --- |
| B-001 | [Positive] Verify API can receive updated order list data | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user do precondition Then user navigate to https://commsult.com/api/v1/order/update Then user enter following request body: {  "driver\_id": "commsult\_best\_driver"  "order\_list": ini\_order\_list\_variable } Then user hit endpoint | Then API returns status code 200 and data available in database |
| B-002 | [Positive] Verify API can receive order confirmation data | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user do precondition Then user navigate to https://commsult.com/api/v1/order/confirm Then user enter following request body: {  "driver\_id": "commsult\_best\_driver"  "data": ini\_data } Then user hit endpoint | Then API returns status code 200 and data available in database |
| B-003 | [Positive] Verify WMS API can receive webhooks (for notification) | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user do precondition Then user navigate to https://WMScommsult.com/api/v1/order/webhook Then user enter following request body: {  "event": "Order update"  "data": ini\_data\_update } Then user hit endpoint | Then API returns 200 and the webhook available in WMS database |
| B-004 | [Positive] Verify backend will send webhook to WMS API after receiving order update data | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user do precondition Then user navigate to https://commsult.com/api/v1/order/update Then user enter following request body: {  "driver\_id": "commsult\_best\_driver"  "order\_list": ini\_order\_list\_variable } Then user hit endpoint | Then API returns status code 200 and webhook will be sent to WMS API and the webhook available in WMS database |
| B-005 | [Positive] Verify backend will send webhook to WMS API after receiving order confirmation data | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user do precondition Then user navigate to https://commsult.com/api/v1/order/confirm Then user enter following request body: {  "driver\_id": "commsult\_best\_driver"  "data": ini\_data } Then user hit endpoint | Then API returns status code 200 and webhook will be sent to WMS API and the webhook available in WMS database |
| B-006 | [Negative] Verify API can't receive incomplete body request data for order confirmation | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user do precondition Then user navigate to https://commsult.com/api/v1/order/confirm Then user enter following request body: {  "driver\_id": "commsult\_best\_driver"  "data":  } Then user hit endpoint | Then API returns status code 422 |
| B-007 | [Negative] Verify API can't receive incomplete body request data for order update | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user do precondition Then user navigate to https://commsult.com/api/v1/order/update Then user enter following request body: {  "driver\_id": "commsult\_best\_driver"  "order\_list": ini\_order\_list\_variable } Then user hit endpoint | Then API returns status code 422 |
| B-008 | [Negative] Verify API can't process unauthorized access for order update | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user don't do precondition Then user navigate to https://commsult.com/api/v1/order/update Then user enter following request body: {  "driver\_id": "commsult\_best\_driver"  "order\_list": ini\_order\_list\_variable } Then user hit endpoint | Then API returns status code 401 |
| B-009 | [Negative] Verify API can't process unauthorized access for order confirmation | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user don't do precondition Then user navigate to https://commsult.com/api/v1/order/confirm Then user enter following request body: {  "driver\_id": "commsult\_best\_driver"  "data":  } Then user hit endpoint | Then API returns status code 401 |
| B-010 | [Negative] Verify WMS API not allowing unauthorized access for webhook | Given user do authorization with token: xxxxxxxxyyyyyzzzzz | Given user don't do precondition Then user navigate to https://commsult.com/api/v1/order/update Then user enter following request body: {  "driver\_id": "commsult\_best\_driver"  "order\_list": ini\_order\_list\_variable } Then user hit endpoint | Then API returns status code 401 |

11.1.2 WMS (notification)

| Test-ID | Test Title | Pre-condition | Steps | Expected result |
| --- | --- | --- | --- | --- |
| A-001 | [positive] User can receive notification when driver update order list in real-time | Given user already login and distribute order to drivers | Given user do precondition Then driver update the order list at 02:00 PM | Then user can see the notification on driver update list with timestamp showing 02:00 PM |
| A-002 | [positive] User can receive notification when driver confirm delivery on updated order list | Given user already login and distribute order to drivers | Given user do precondition Then driver update the order list at 02:00 PM Then user will see update order list notification Then driver confirm changed/adjusted order card at 02:02 PM | Then user can see the confirm notification with timestamp showing 02:02 PM and total deeliveries completed by driver |
| A-003 | [Negative] User will not receive notification when driver drag-and-drop order card in same position | Given user already login and distribute order to drivers | Given user do precondition Then driver drag-and-drop order 004 to be top of order list and back to be below order 003 Then driver drop the order 004 below order 003 | Then user will not see notification from driver mobile app |
| A-004 | [Negative] User will not receive notification when driver drag-and-drop order card in same position | Given user already login and distribute order to drivers | Given user do precondition Then driver hold and drag order 003 card to be below 005 and back to below order 002 Then driver drop the order 003 card below order 002 | Then user will not see notification from driver mobile app |
| A-005 | [Negative] User will not receive notification when driver drag-and-drop completed order | Given user already login and distribute order to drivers | Given user do precondition Then driver confirms order card 001 Then driver hold and drag order card 001 to be below order card 003 Then driver drop the order card 001 below order card 003 | Then user will not see notification from driver mobile app |

11.1.3 Mobile app (drag-and-drop feature)

| Test-ID | Test Title | Pre-condition | Steps | Expected result |
| --- | --- | --- | --- | --- |
| F-001 | [Positive] User can update order list on mobile after receiving order list from analyst | Given user already login and already receive order list from analyst | Given user do precondition Then user hold and drag order 001 card to be below order 005 card Then user drop order 001 card below order 005 card | Then user will see the order list has changed with order 001 below order 005 |
| F-002 | [Positive] User can confirms delivery card on updated card list | Given user already login and already receive order list from analyst | Given user do precondition Then user change the position of order 005 card to be above order 001 card Then user confirms delivery for order 005 | Then user can click confirm button for order 005 delivery |
| F-003 | [Negative] User can't update order list if the card dropped outside the list table | Given user already login and already receive order list from analyst | Given user do precondition Then user hold and drag order 001 card to be outside the order card list zone Then user drop the order 001 card outside the zone | Then user will see the order list unchanged |
| F-004 | [Negative] User can't update completed order list position in order list | Given user already login and already receive order list from analyst | Given user do precondition Then user confirms order card 001 Then user hold and drag order card 001 to be below order card 003 Then user drop the order card 001 below order card 003 | Then user will see the order list unchanged and completed order card returns to it position |
| F-005 | [Negative] User can't update order list if the card dropped in exact same position | Given user already login and already receive order list from analyst | Given user do precondition Then user hold and drag order 003 card to be below 005 and back to below order 002 Then user drop the order 003 card below order 002 | Then user will see the order list unchanged |

11.1.4 Regression test scripts

Following legacy features will be tested:

1. Login
2. Order creation by analyst
3. Stops sorting for order creation
4. Order distribution to driver
5. Receiving order on driver mobile app
6. Delivery confirmation feature on driver mobile app
7. Delivery confirmation notification on WMS
8. End-to-end usual business flow

11.2 Test Execution

The **manual** test cases will be managed in test rail, but first we will use excel before importing it to test rail for documentation and archive with their respective graph. **Usability** testing will be conducted by gathering 50 or more participants of driver mobile app user, by using maze. The test will begin with the question about the participant background (new employee or not, and how long the participant has been working as our driver. Then the participant will be tested with several scenario to update order with/without encountered error scenario. Then the result would be automatically calculated and displayed in result tab in maze platform. The **regression** test will be using katalon for mobile test and katalon ops for cloud regression test.

11.3 Defect Tracking

The process of managing bugs/defects will happen on JIRA. When reporting, we will create a card consisting title, description, severity, priority, status, attachment, assignee, and tester assignee. The defect has following life cycle:

* Identification – First encounter and troubleshooting
* Reporting – creating card in JIRA and communicating with team
* Prioritization – assigning priority level
* Assignment – after discussing the bugs with responsible developer, try to assign the developer (could be new or responsible developer) and tester.
* Fixing – Developer working to fix defect/bug
* Verification/testing – After done fixing, QA starts reproduce the bug/defect
* Deploying hotfix – After passed the verification, hotfix deployed
* Closure – closing the card in JIRA

# 12. Approval and Sign-off

There is following approval and sign-off process for this document, which consist:

1. Draft review. Reviewing draft will be presented to stakeholders including PM, Developers, and QA team (all parties).
2. Revision (if there is any). Changing and adjusting current draft to be aligned with stakeholders and our team needs and suggested coverage.
3. Final review. Stakeholders will review the final draft and after approved, the testing will be started soon.
4. Approval signature. All of stakeholders approve the test plans, which means they agree with scope, objectives, test approach, deliverables, and every plan inside the document.