

Neptune DXP (Open Edition) Test Plan

Revision History

Date	Version	Author	Description
18.05.2022	1.0	Rommel Lamanilao	Test plan for Neptune DXP Open Edition 22-LTS release

Table of Contents

1. INTRODUCTION.....	3
1.1. Overview	3
2. OBJECTIVES AND TASKS.....	4
2.1. Objectives.....	4
2.2. Tasks.....	4
3. SCOPE AND LIMITATIONS	5
3.1. Scope	5
3.2. Limitations	6
4. TESTING STRATEGY	7
5. TEST CRITERIA	8
5.1. Suspension Criteria	8
5.2. Exit Criteria.....	8
6. RESOURCE PLANNING	9
6.1. System Resource	9
6.2. Human Resource.....	9
7. TEST CASE DESIGN	10
8. TEST ENVIRONMENT.....	11
8.1. Test Server Setup.....	11
8.2. Device	11
8.3. Browser	11
9. SCHEDULE AND ESTIMATION.....	12
9.1. Test Estimation.....	12
9.2. Test Schedule.....	12
10. RISKS AND MITIGATIONS	13
11. TOOLS.....	14
12. BUG REPORTING	15
12.1. Bug Priority.....	15
12.2. Bug Category.....	15

1. INTRODUCTION

1.1. Overview

Neptune DXP — LTS (long-term support) version offers all the features rolled out in one production-ready release. Prior to its actual release, several tests must be executed to ensure that at least all major functionalities are working. This test plan is created to guide the QA Team for the entire testing process.

Also, this document is intended for the testing process of **Neptune DXP — Open Edition 22-LTS** version.

2. OBJECTIVES AND TASKS

2.1. Objectives

The person in charge of the tests should pay attention to this objective before actual tests are performed. The initial release notes can be used as reference to help meet these objectives.

The objectives of the test should be:

- that all new features are working as expected
- that all bug fixes should be tested and verified
- that all major and minor functionalities should be working
- that all other tests defined should PASS

2.2. Tasks

The activities involve in this test plan are the following:

- Determine the scope of the test — tests to be performed and NOT to be performed
- Document the Test Strategy
- Decide the test criteria — suspension and exit criteria
- Plan the test resources
- Plan the test environment
- Plan when and how to test
- Deliver test results as part of the execution

3. SCOPE AND LIMITATIONS

3.1. Scope

Testing will mostly cover the following:

3.1.1. New Features

Key contact persons are provided on each feature. Person who is doing the specific feature must refer to the key contact for the procedure on how to test. More details on how to test these can be found in the [Test Management](#) tool.

- **WCAG A & AA Compliance**

Use WAVE extension in Chrome to highlight any errors

Key Contact: John Cocks

- **Connectors (API, Server Script & Table Support)**

- An option to clean up field catalog
- Generating of mapping automatically
- Filtering of connectors by type
- ...more can be found in the above mentioned tool

Key Contact: John Cocks

- **Highcharts Library Update**

Change of highchart version to 10.0.0 and test if it works on the following:

- Launchpad app
- API Trace
- Launchpad Trace
- Test Run
- Load Test
- Basic test in App Designer
- ...more can be found in the above mentioned tool

Key Contact: Muhammad Adeel

- **Custom Components**

- Create a custom component
- Set an icon for the custom component
- Add/Remove script on custom component
- ... more can be found in the above mentioned tool

Key Contact: John Cocks

- **Advanced TypeScript Support**

- Load definitions from other apps

- Load definitions from launchpad
- Export source maps correctly
- Load completions from custom components
- Everything should work in App Handler V1 and V2
- ... more can be found in the above mentioned tool

3.1.2. Basic usability and accessibility

3.1.3. Re-verification of bug fixes

The list of GitHub issues with fixes can be found in the tool mentioned above.

3.2. Limitations

- Performance Testing
- Stress Testing
- Database Testing
- GitHub changes and commits not testable from tester's perspective and only developers can verify such as server configuration files

**Not on regular basis but will be performed if required*

4. TESTING STRATEGY

A separate document is created for this section which you can find it here:

[Neptune DXP Open Edition - Test Strategy.docx](#)

5. TEST CRITERIA

5.1. Suspension Criteria

If there is at least one major functionality that is not working, for example, unable to create a mobile build then suspend the release of the build until the development team fixes it.

5.2. Exit Criteria

All test cases are performed with a mandatory of at least 80% PASS rate

6. RESOURCE PLANNING

6.1. System Resource

Resources	Description
Server	Installation of Neptune DXP Open Edition on a cloud
Automated test scripts	These scripts are developed using TestCafe . Needed to perform the regression testing.
GitHub access	GitHub is used for bug tracking. If necessary, an access must be provided to all members of the QA Team

6.2. Human Resource

Resources	Tasks
Test Manager <ul style="list-style-type: none">Rommel Lamanilao	<ul style="list-style-type: none">Manages the entire testing processCreates and maintains the test plan and test strategy
Software Tester <ul style="list-style-type: none">Luis GonzalesCarlos Daniel Silva Reis PiresAndré CarrilhoDavid CarrilhoLloyd Trevarthen	<ul style="list-style-type: none">Executes the test casesReports bugs/issues found to GitHubVerifies bug fixesCreates and maintains test cases
Automation Tester	<ul style="list-style-type: none">Develops and maintains automated test scriptsExecutes automated test scriptsManages end-to-end tests on CI/CD pipeline (CI-Nightly)

6.2.1. Test Responsible

Here we define the list of testers who will be assigned to perform the testing for Neptune DXP — Open Edition

- Rommel Lamanilao
- Carlos Daniel Silva Reis Pires
- Lloyd Trevarthen

7. TEST CASE DESIGN

Test cases should be defined to determine if different features within a system are performing as expected and to confirm that the system satisfies all related standards, guidelines, and customer requirements. This process can also help reveal errors or defects within the system.

Neptune DXP Open Edition has an app called "[Test Unit](#)" where you can write test cases.

8. TEST ENVIRONMENT

The following should be setup to ensure software testing process success.

8.1. Test Server Setup

Install the Neptune DXP Open Edition on a local server or on the cloud.

Responsible: Thomas Bergheim and Ian Sager

8.2. Device

Tests will be performed on the following devices:

- Desktop
- Tablet
- Phone

8.3. Browser

The following browsers should be installed:

- Google Chrome
- Mozilla Firefox
- MS Edge
- Safari

9. SCHEDULE AND ESTIMATION

9.1. Test Estimation

Testing	Estimate Effort (%)
Smoke Test	10
Integration Test	40
System Test	50
TOTAL	100

9.2. Test Schedule

	Planned Date	Actual Date
Test started date		
Test completed date		

10. RISKS AND MITIGATIONS

Risks	Mitigations
Lack of test resource availability	Facilitate hiring or loan people from another department
Shortage of time to cover all the scope	<ul style="list-style-type: none">• Focus on the new features and major changes• Focus on critical areas
Defects are found at a late stage and consumes time to resolve	<ul style="list-style-type: none">• Defect management plan is in place to ensure prompt communication and fixing of issues

11. TOOLS

Tools	Usage
GitHub	For reporting of issues found during the testing and for requesting for a new feature
TestCafe	For creating automated test scripts
WAVE Evaluation Tool	It is an extension for Chrome browser that provides visual feedback about the accessibility of the web content. This is useful for testing the WCAG compliance feature

12. BUG REPORTING

All encountered issues and feature requests should be reported to GitHub.

12.1. Bug Priority

Bug priority refers to how urgently a bug needs to be fixed. When creating a bug report, one must provide a priority on it. Although GitHub does not have a field for this priority, labels have defined instead.

Levels of bug priority defined as labels in GitHub:

- **Low:** Bug can be fixed at a later date. Other more serious bugs take priority
- **Medium:** Bug can be fixed in the normal course of development and testing
- **High:** Bug must be resolved at the earliest as it affects the system and renders it unusable until it is resolved

12.2. Bug Category

Like in bug priority, GitHub does not have category field as well so we will use labels instead. Category refers to the module or component where the bug is found. This is not a mandatory field but it is nice to have for easy filtering of the issue list.

These are the commonly used labels for categories:

- Use this label for general category particularly for smaller components/applications
 - Cockpit:
- For bigger components/applications:
 - Launchpad
 - App Designer
 - App Editor
 - Script Editor
 - Adaptive Designer
 - Deployment
 - Custom Component