Neptune DXP 23 - Open Edition Test Plan

Revision History

Date	Version	Author	Description
28.09.2023	1.0	Rommel Lamanilao	Test plan for Neptune DXP Open Edition 23-LTS release

Table of Contents

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 23-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

More details on how to test these can be found in the Test Management tool.

- **OpenAI Chat**
  - Parameter validation in User Settings
  - Popover documentation
  - Chat interface

- **Mobile Client**
  - Filter in the build list
  - Android engine version
  - Adding of **targetSdkVersion**
  - Setting of **minSdkVersion** and **maxSdkVersion**

- **Cockpit**
  - Custom cockpit tiles
  - Pagination in the user selection
  - Customizing visible columns
  - Artifact search results
  - Favorite apps
  - Basic test in App Designer
  - Marketplace featured items

- **Launchpad**
  - PWA install dialog
  - Lazy/offline loading of images

- **System Settings**
  - Authentication to SMTP servers
  - Response type field for OpenID authentication
- **Table Definition**
  - Import JSON data for definition of structure
  - Generate API based on table
- **App Designer**
  - Create versions on activate
  - Context menu items
  - Middle clicking on the tab
  - Toolbar aggregation for sap.f.cards.Header
  - Support for AvatarGroup, ProductSwitch, SidePanel, DynamicDateRange

- Support for sap.m.MessageView
- …more features can be found in the Test Management 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:

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 Playwright. 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 • Rommel Lamanilao	• Manages the entire testing process • Creates and maintains the test plan and test strategy
Software Tester • Luis Gonzales • Carlos Daniel Silva Reis Pires • André Carrilho • David Carrilho • Daniel Vieira • Sofia Natálio • Cezar Stefan Strugariu • Miguel Coelho • Levi Bailey	• Executes the test cases • Reports bugs/issues found to GitHub • Verifies bug fixes • Creates and maintains test cases
Automation Tester	• Develops and maintains automated test scripts • Executes automated test scripts • Manages 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 23-LTS — Open Edition

- Rommel Lamanilao
- Carlos Daniel Silva Reis Pires
- Cezar Stefan Strugariu
- Levi Bailey
- Miguel Coelho

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: Rommel Lamanilao

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	02-October-2023	
Test completed date	31-October-2023	

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	• Focus on the new features and major changes • Focus on critical areas
Defects are found at a late stage and consumes time to resolve	• 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
Playwright	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