Neptune DXP
Test Strategy

Revision History

Date	Version	Author	Description
27.05.2022	1.0	Rommel Lamanilao	Initial
28.09.2023	2.0	Rommel Lamanilao	Update image used in the intro page and tool used of the test automation

Table of Contents

1. INTRODUCTION

1.1. Overview

This is the Test Strategy for Neptune DXP. This document shall be completed and used by the project test team to guide how testing will be managed for this project. The test effort will be prioritized and executed based on the project priorities as defined in the Test Plan. This document may be revised as the project progresses. The Test Manager, Chief Architect and Engineering Head shall review and approve the final version of this document.

1.2. Purpose

The purpose of this Test Strategy is to help testers and the rest of the testing team define an approach to test Neptune DXP. This approach will eventually lead to an excellent quality of the application.

1.3. Definitions and Acronymns

Application Under Test (AUT)

When the design and coding of the module or application is done, it will be forwarded to the testing team for them to test it. That application will then be called AUT or Application Under Test.

Long Term Support (LTS)

A kind of support that throughout the lifetime of a release there is a commitment to update, patch and maintain the software.

GitHub

A cloud-based service that helps developers store and manage their code, track for issues and control changes to their code.

Test Case

A specific set of test data along with expected results for a particular test objective.

Test Strategy

Describes the scope, approach, resources, and schedule for the testing activities of the project

Playwright

a cross-browser framework, provides end-to-end automated testing of web applications using the same API..

2. SCOPE AND LIMITATIONS

2.1. Scope

Testing will mostly cover the following:

- Major functionalities
- New features
- Basic Usability and Accessibility
- Verification of Bug Fixes
- Regression tests

2.2. 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*

3. TESTING APPROACH

3.1. Fundamental Test Process

3.1.1. Test Planning

Define detailed testing information such as:

- Scope of testing
- Test objectives
- Resources and test environments
- Test tools
- Types of testing
- Schedule

3.1.2. Test Analysis and Design

This stage involves the following activities:

- Design test cases based on the test objectives
- Write test scripts and test data
- Design test automation scripts
- Evaluate the testability of the software based on requirements

3.1.3. Test Implementation and Execution

Activities involve in this stage:

- Group test cases together that have the same set of behaviours
- Execute the developed test cases and scenarios
- Execute necessary automation test scripts
- Re-execute the tests that are failed to confirm the bug fixes
- Verify the actual and expected results of the test cases
- Report any discrepancies in the results

3.1.4. Evaluating Exit Criteria and Reporting

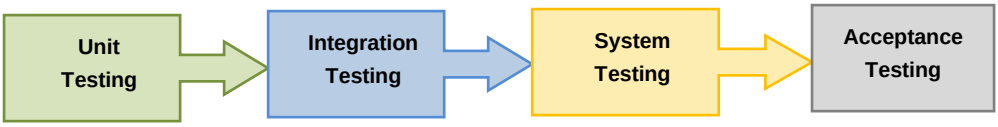
Exit criteria is evaluated on the following:

- Executed more test cases with certain pass percentage
- Bug rate is low
- Deadlines achieved
- No critical issues

3.1.5. Test Closure

Test closure happens when the exit criteria is met, and all fixed issues have been verified as fix.

3.2. Testing Levels



3.2.1. Unit Testing

Purpose	To test the individual configuration to ensure that they function according to the detailed technical specification. A module or component is tested in isolation
Development Phase	Development and Testing
Test Scope	All configurations, code validation...
Responsible	Development Team
Exit Criteria	• All unit test cases are completed successfully • All source code is unit tested • No outstanding critical issues • All outstanding issues are reported to GitHub

3.2.2. Integration Testing

Purpose	To validate the full operability of interconnected functions, methods or objects within a functional area
Development Phase	Development and Testing
Test Scope	All functional tests mentioned in the Scope section
Responsible	QA Team
Exit Criteria	• Test case execution completed with 90% passed • All discovered issues are reported in GitHub • No outstanding "**showstopper or blocker**" issues • All test results have been documented

3.2.3. System Testing

Purpose	To test the complete integrated application
Development Phase	Development and Testing
Test Scope	• Regression testing • End-to-end testing
Responsible	QA Team
Exit Criteria	• Test case execution completed with 100% passed • All issues are reported in GitHub • No outstanding "**showstopper or blocker**" issues • All test results have been documented

3.2.4. User Acceptance Testing

Purpose	To test the complete, end-to-end business processes to verify that the implemented solution performs the intended functions and satisfies the business requirements
Development Phase	Final Prep / Implementation
Test Scope	• Full regression • UAT
Responsible	Business Users
Exit Criteria	• Acceptance tests must be completed with a pass rate of 98% • No outstanding "**showstopper or blocker**" issues • All test cases have been completed • UAT test summary report documented and approved

3.3. Test Types

During the QA testing, different types of testing can be performed, and these are:

Type	Objective
Smoke Testing	Whenever there is a new build, the QA Team will first determine the major functionality in the application to check for "***showstopper***" or "***blocker***" issues
Functional Testing	To determine if a piece of component is working in accordance with its requirement
Feature Testing	To verify that all newly added features are working
Regression Testing	• To examine the functionality of the entire system after upgrade • To identify any errors in the existing functionalities • To ensure that the implementation of new features does not interfere with existing components • To verify that no side effects are created after the bug fixes
End-to-End Testing	To test the entire product from beginning to end to ensure the application flow behaves as expected

3.4. Roles and Responsibilities

Role	Responsibility
Test Manager	• Elaborate test plans and test strategies • Manage and coordinate test team activities
Software Tester	• Create manual test cases • Execute manual test cases and report for any defects found

Automation Tester	• Prepare and execute automated test cases • Provide test execution reports

3.5. Issue Reporting

All discovered issues should be reported in GitHub.

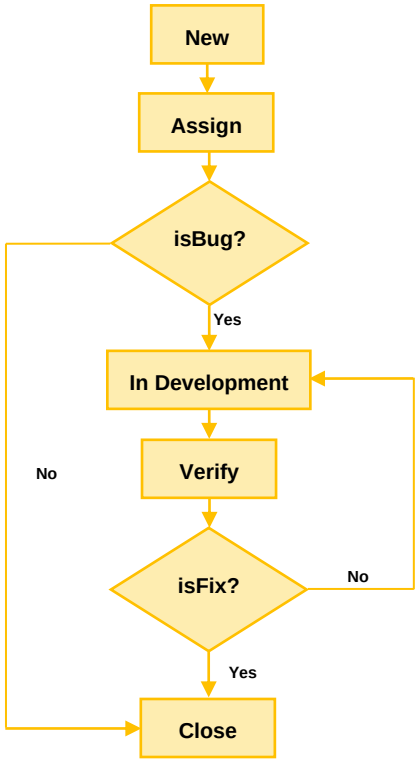
3.5.1. GitHub Links

The following links should be used according to Neptune product

- **Neptune DXP — SAP Edition**
  https://github.com/neptune-software/dxp-sap-edition/issues

- **Neptune DXP — Open Edition**
  https://github.com/neptune-software/planet9/issues

3.5.2. Issue Management Process



New	Intial status of the issue is "**Open**" and marked as "**Bug**" or "**Feature Request**"
Assign	Issue is assigned to a developer. Developer can also self-assign the issue
In Development	If issue is indeed a bug then developer starts to fix the issue
Verify	The developer marks the issue as "**Verify**" once the fix is ready to be tested and assign it to the QA Team
Reject / Close	Tester verifies the fix and then mark the issue as "**Verified**" otherwise "**Fix Not Working**" and assign it back to the developer

3.5.3. Issue Report Information

Issue ID	Unique identification number of the issue. GitHub auto-generates this ID after issue is submitted
Brief Title	Short but descriptive title of the issue
Assignees	Person responsible for the development or for testing
Description	Detailed information about the issue should contain the following: • Description/Summary • Version • Application • Steps to Reproduce • Actual Behaviour • Screenshots/Screen Recording (nice to have) • Expected Behaviour
Labels	Labels are used for easy filtering of the issues. These are the commonly used labels: • Bug — use this to report a bug found in the system • Feature Request — for requesting a new feature • Cockpit — for issues related to cockpit • Launchpad — for issues related to launchpad • App Designer — for issues related to App Designer
Milestone	Refers to the target build release where the fix is included. Leave this field blank for all launchpad and adaptive template related issues.

4. TEST ENVIRONMENT

4.1. Test Server Setup

The server to be used for the test is normally being setup and configure by the Chief Architect. A URL will be provided once it is complete.

4.2. Devices

Tests should be executed on the following devices:

- Desktop
- Tablet
- Mobile Phones

4.3. Browsers

Tests should be performed at least on the following browsers:

- Google Chrome
- Mozilla Firefox
- MS Edge
- Safari

5. TESTING TOOLS

The following tools will be used for testing:

Process	Tool
Test Case Development	Test Unit — Neptune DXP Open Edition
Defect Tracking Tool	GitHub
Automated Test Execution	Playwright
CI/CD Tool	GitHub Workflow
Source Control	GitHub

6. RELEASE CONTROL

Activities	Person Responsible
QA plan and test cases have been updated	Test Manager
QA plan has been completely carried out	Test Manager
All discovered issues have been reported to GitHub	Tester
All fixed issues have been verified as fixed	Tester
QA team is satisfied with the release	QA Team

7. RISK ANALYSIS

Risks	Mitigations
The shortfall of test resources	Facilitate the hiring or training process
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