**Luma website project**

Test Strategy

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Description** |
| 16.11.2024 | 1.00 | Oleksandr Burenok | Essential text for test strategy. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Scope…………………………………………………………………………..……..3

2. Test Approach……………………………………………………………..…………4

3. Test Environment………………………………………………………..…………..5

4. Testing Tools……………………………………………………………..…………..6

5. Release Control ………………………………………………………..……………7

6. Risk Analysis. ……………………………………………………………..…………8

7. Review and Approvals. …………………………………………………..…………9

1. **Scope**

The scope of the testing is to cover all functionality of the developed website. The website should be checked at least on the Chrome browser using manual testing.

For the future maintenance and development automation testing must be implemented. Automation should cover tests like CRUD and CoreFlow.

Time allocated for all test activities is 1 year and should be finished 20th of November 2025. All test documentation should be saved and delivered to the customer.

The test strategy document can be approved by Product Manager/Product Owner.

The test strategy document can be modified by Test Manager only.

1. **Testing Approach**

Test objects

* Verify and validate that functionality works according to requirements.
* Provide confidence for the business owner that the product satisfies business needs, etc.
* Establish a process to improve product quality.

## Testing Types

* Unit tests validate the smallest components of the system, ensuring they correctly handle known input and outputs.
* Functional tests verify scenarios that cover business needs.
* System tests should be covered by manual testing once, right after development. The automation tests should cover regression testing.

The main source of the website requirements are requirements provided by the Product Owner.

1. **Test Environment**

One team has one manual test engineer and one automation test engineer, so every team should have:

* Two environments for automation testing (Linux for running website and Windows for running tests) with nightly updates and nightly test run.
* Two environments for automation engineer (Linux for running website and Windows for creating auto tests).
* One environment for manual testing (can be shared with developers).

General test data (GTD) will be defined by a separate team and uploaded to the website database. GTD should be the the same for all test environments.

1. **Testing Tools**

To perform manual and automation testing for current project, next tools/technologies will be used:

* Python
* Selenium
* Pytest
* PyCharm
* MS Excel
* Jira (comercial)
* Jira Adaptavist (comercial)
* Jira Fischeye
* Chrome
* Firefox
* Edge
* Google Cloud (comercial)
* VPN client

1. **Release Control**

To provide well done release next statements should be satisfied:

* All test cases assigned to the story are executed.
* All sub-tasks are closed.
* All story-defects are closed.
* There are no any opened major defects (Severity 0 or 1).
* Functional tests must be created, reviewed and executed. All functional tests are passed.
* Unit and Functional tests are automated via a separate JIRA Ticket or planned to be automated.
* The user stories are demoed to the Product Owner.
* The Product Owner has approved all user stories.

1. **Risk Analysis**

Impact analysis can be used or a general agreed approach within the team. Be sure that next risks are discussed with team:

* Is the Epic/User Stories well documented?
* Has there been a discussion with a customer before the first sprint starts?
* Is the team working in an area that they have never worked before?
* Are there any long term holidays planned during the next sprints (holidays)?
* Is there new technology involved?
* Are all necessary tools provided?

If any of the risks happened during development, it should be:

* classified (relates to quality, performance, usability, etc…)
* define level of the risk probability (high - 50%-90%, medium - 15%-50%, unlikely - 0%-15%)
* define level of impact. Major - functionality severely damaged. Moderate - significant time/resources required. Minor - some disruptions possible.

All new risks should be classified and discussed with the Product Owner or business team, to define the solution and when it will be implemented.

1. **Review and Approvals**

All these activities are reviewed and sign off by:

* System Administration Team -
* Project Management Team -
* Development Team -
* Business Team-