Simulation tool for computing platform

TEST PLAN

Version <1.0>

TABLE OF CONTENTS

1 Introduction

1.1 Purpose of The Test Plan Document

This Test Plan document documents and tracks the necessary information required to effectively define the approach to be used in the testing of the project's product. The Test Plan document is created during the Planning Phase of the project. Its intended audience is the IT support team.

2 Compatibility Testing

2.1 Test Approach(s)

The principal compatibility problems we can encounter with this kind of software (web-application) is the web browser compatibility, since browsers are already compatible on every kind of computing environment.

2.2 Test Pass / Fail Criteria

The pass criteria will be the successful loading of the web page in the browser. The test will be considered as failed if the web app shows weirdly on the web browser.

2.3 Test Deliverables

The deliverable will be a table, present in the test results document.

2.4 Test Environmental / Staffing / Training Needs

To run this test, necessary software are some web browsers (Internet Explorer, Mozilla Firefox, Google Chrome, Safari) in their latest versions.

This test requires to launch the web application locally, with Node and Angular CLI, and a web browser.

3 Conformance/Functional Testing

3.1 Test Approach(s)

These tests will confirm all the features announced in the requirements.

3.2 Test Pass / Fail Criteria

A test will be passed if the feature is available and working as mentioned in the requirements. It will fail if not.

3.3 Test Deliverables

The deliverable will be a table, present in the test results document. It will contain results of Karma testing tool outputs.

3.4 Test Environmental / Staffing / Training Needs

This test requires to launch the web application locally, with Node and Angular CLI, and a web browser. It needs Karma testing tool, which is already in Angular CLI.

4 Load Testing

4.1 Test Approach(s)

The idea of this test is usually to measure the response of an application, a server or a database under a big amount of users and then of requests. Here, the user will always be unique. However, performance testing will give us better results (qualitatively and quantitatively) with the reaction of the software under big simulation requests (lot of users, long simulation, etc.)

5 Performance Testing

5.1 Test Approach(s)

The idea of those tests is to compare the performance of the software under differents work loads.

5.2 Test Pass / Fail Criteria

Test will be considered as passed if the differences of performance are similar whatever to work load is. However, it will fail if these differences are too important.

5.3 Test Deliverables

The deliverables for this test will be different outputs of the system, each corresponding to different configurations of the HPC system and its simulated users.

5.4 Test Environmental / Staffing / Training Needs

This test requires to launch the web application locally, with Node and Angular CLI, and a web browser.

6 Regression Testing

Considering the fact that the software will be used only during a small period of time, and due to its small number of dependencies, no regression testing will be planned. However, if the IT staff decides to use it again to run simulations, regression tests must be planned and performed.

7 Unit Testing

7.1 Test Approach(s)

The idea is to test if parts of the software works correctly on their own. These tests will be done using Karma.

7.2 Test Pass / Fail Criteria

Tests will be considered as passed/failed if Karma util does.

7.3 Test Deliverables

Test deliverables will be screenshots of Karma interface.

7.4 Test Environmental / Staffing / Training Needs

This test requires to launch the web application locally, with Node and Angular CLI (contains Karma utils), and a web browser.

8 User Acceptance Testing

8.1 Test Approach(s)

Beta testing will be provided by the software developers and the IT staff, and also during performance testing. As a user acceptance testing for a open source software, issues or features request will be defined and managed directly on project repository on GitHub.