Test Plan

Lezioni alla pari

April 19, 2020

Team Members

Ovidiu Andrioaia
David Cirdan
Luciano Mateias
Zhiyang Xia

Document Control

Change History

Revision	Change Date	Description of changes
V1.0	04/19/2020	Initial release

Document storage

This document is stored in the project's GIT repository at: https://github.com/KilliKrate/Software-Documentation-G6/blob/master/docs/Test%20Plan/index.md

Document Owner

Group 6 is responsible for developing and maintaining this document.

Table of Contents

Introduction

This document outlines the test plan for the Lezioni alla Pari application which, as described in the Software Requirements Specification document, provides an intuitive and easy to use platform that enables closed groups, like the departments of a company or the classes of a school, to share informative data in the form of lessons and quizzes.

The test plan outlined in this document provides a strategy for analyzing and uncovering potential issues in the program, like bugs or architecture faults, which can tamper with the user experience or general usability of the program. The testing activies described will ensure that the systems meets its requirements and target features.

Back to Top

Items and Features Tested

Items that will be tested during the testing phase by the project's developer team include the Login Application, through which individuals can sign-in and sign-up to the service, and the Lezioni alla Pari Client. The features tested include, but are not limited to:

Login Application

- The ability for an individual to sign up to the service (1.1)
- The ability for a user to log into the service (1.2)
- The ability for a user to change his user settings, like resetting his password (1.3)

Lezioni alla Pari Client

- The ability for a user to view the courses, topics and lessons/quizzes list (2.1)
- The ability for a user to create courses, topics and lessons/quizzes for a certain topic (2.2)
- The ability for a user to delete an existing lesson or quiz (2.3)
- The ability for a user to view a lesson (2.4)
- The ability for a user to complete quizzes (2.5)
- The ability for a user to modify a lessons's contents (2.6)
- The ability for a user to modify a quiz's questions and answers (2.7)

Back to Top

Items not Tested

There are features that will not be included in the current testing procedure. They have yet to be implemented, and because of that they are not available for testing. These features include, but are not limited to:

- Administrative features that include adding or removing an user
- Functions tied to one's email, like email confirmation and preferences
- Permission functions, like adding a co-instructor/moderator to a course or inviting students to a private course (as of now, all courses are public by default)

Back to Top

Approach

The testing approach will be conducted by the "Lezioni alla Pari" developement team during the alpha phase. The team will use the application and try to do things a normal user would do, like logging in, reading a lesson and maybe do a quiz, but also perform a series of low-level tests, to ensure the integrity of the code logic (white-box testing). The team will also try to break application by inserting invalid data or trying to use the software in an improper manner.

During the beta phase, the software will be rolled out to close friends and relatives to test. By obtaining an outsider's approach (black-box testing), we will be able to discover bugs that also originate from improper use that a developer team might not have even thought about. All issues found will be fixed before the final release. After release, testing will be lead by the internal developement team, and scheduled before an update, in order to ascertain that the features work as intended.

Internal architecture testing and stress tests will also occour to determine the size of the workload the servers can manage. This testing will also ensure that the system fullfills the requirements that have been agreed upon with the client.

Back to Top

Test Reports

After an issue has been found, the testers will write a report, in order to track the amount and severity of issues with the software, which will be submitted to the developement team (in the case of external testers) or directly shared with the key roles responsible for fixing the issue. These reports are split up between three main categories: incidents, defects and changes.

Incidents

Incidents are problems related to a degradation in the existing software platform: this can happen in the form of high loading times, which indicate a network or server problem, or errors in the new (or most often old) version of the browser the platform is run on (this is not our case, as we have our own desktop client).

Defects

Defects are problems associated with a direct "break/fix" problem, like a bug in the software, which can often be fixed with a software update.

Changes

Changes are problems that result after a change has been made in the software: for example, an update of the graphical user interface might upset some users, which were already used to the old layout, or maybe the new interface is less accessible with people with screen readers, which impairs their ability to efficiently navigate the platform's contents.

Back to Top

Item Pass/Fail Criteria

The features that the software provides, and the minimum requirements associated with them, have already been described in the Software Requirements Specification document. All the features present will have to comply to those requirements, as specified by the client. Features that fail the testing procedure, or do not meet the requirements specified by the client will be documented via a test report and turned over to the developement team for investigation and revision.

Back to Top

Test Deliverables

In addition to the Test Plan, test deliverables have been included in the Test Specification, which outlines the specific test cases and expected results of each test. The Test report, as mentioned earlier, will be comprised of Incidents, Defects and Changes

Back to Top

Test Goals

The goals of this testing exercise include finding various user situations and testing against these situations in order to discover any errors that may cause failures. The ultimate goal is to ensure that the team delivers an error-free functioning system.

Back to Top