QA Team “Apple”

Telerik Academy 2014

**Evaluation & Teamwork Modules**

**Test Plan**

Version 5.0

November 25, 2014

# Table of Contents

[Table of Contents 2](#_Toc404323465)

[1. VERSION HISTORY 4](#_Toc404323466)

[2. INTRODUCTION 5](#_Toc404323467)

[2.1. Project Background 5](#_Toc404323468)

[2.2. Objectives 5](#_Toc404323469)

[2.3. Testing Strategy 5](#_Toc404323470)

[2.4. Scope 6](#_Toc404323471)

[2.5. Reference Material 6](#_Toc404323472)

[3. FEATURES TO BE TESTED 6](#_Toc404323473)

[3.1. AdminUI 6](#_Toc404323474)

[3.1.1. Evaluation modules 6](#_Toc404323475)

[3.1.2. Teamwork modules 7](#_Toc404323476)

[3.2. Front-end UI 7](#_Toc404323477)

[4. FEATURES NOT TO BE TESTED 8](#_Toc404323478)

[5. TESTING APPROACH 8](#_Toc404323479)

[5.1. Integration Testing 8](#_Toc404323480)

[5.2. System Testing 9](#_Toc404323481)

[5.2.1. Functional Testing 9](#_Toc404323482)

[5.2.2. Non-functional Testing 9](#_Toc404323483)

[5.2.3. Regression Testing 10](#_Toc404323484)

[5.3. Acceptance Testing 10](#_Toc404323485)

[6. MEASURES AND METRICS 10](#_Toc404323486)

[7. TESTING CRITERIAS 12](#_Toc404323487)

[7.1. System TestEntrance / ExitCriteria 12](#_Toc404323488)

[7.1.1. Entrance criteria 12](#_Toc404323489)

[7.1.2. Exit criteria 12](#_Toc404323490)

[7.2. Pass / Fail Criteria 13](#_Toc404323491)

[7.3. Suspension / Resumption Criteria 13](#_Toc404323492)

[7.3.1. Suspension Criteria 13](#_Toc404323493)

[7.3.2. Resumption Criteria 13](#_Toc404323494)

[7.4. Approval Criteria 13](#_Toc404323495)

[8. TESTING PROCESS 14](#_Toc404323496)

[8.1. Test Deliverables 14](#_Toc404323497)

[8.2. Testing Tasks 14](#_Toc404323498)

[8.3. Responsibilities 14](#_Toc404323499)

[8.4. Schedule 14](#_Toc404323500)

[9. ENVIRONMENTAL REQUIREMENTS 15](#_Toc404323501)

[9.1. Hardware 15](#_Toc404323502)

[9.2. Software 16](#_Toc404323503)

[9.3. Access 16](#_Toc404323504)

[9.4. Tools 16](#_Toc404323505)

[10. RISKS AND MITIGATION 17](#_Toc404323506)

[11. PLAN APPROVALS 17](#_Toc404323507)

# VERSION HISTORY

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Reason for Change | Revised By | Date |
| 1.0 | Initial Test Plan | Team Apple | 26/9/2014 |
| 2.0 | Major rework after the first review | Team Apple | 3/10/2014 |
| 2.1 | Minor changes | Team Apple | 10/10/2014 |
| 3.0 | Review by Team Redcurrant | Team Apple | 30/10/2014 |
| 4.0 | Performance tests inclusion | Team Apple | 14/11/2014 |
| 4.1 | Test plan is reworked because Performance plan is in separate document. | Team Apple | 21/11/2014 |
| 5.0 | Final adjustments. | Team Apple | 25/11/2014 |
|  |  | Choose an item. |  |
|  |  | Choose an item. |  |
|  |  | Choose an item. |  |
|  |  | Choose an item. |  |
|  |  | Choose an item. |  |

# INTRODUCTION

This document describes the procedure and expectations for testing of “Evaluation” and “Teamwork” modules in the student system being developed by Telerik Academy institution.

## Project Background

The existing system allows the students and administrators to evaluate homework, exams, teammates, etc. in courses they participate. This project is intended to provide testing for the modules “Evaluation” and “Teamwork” in the [Telerik Academy Student System (TASS)](http://test.telerikacademy.com/).

## Objectives

The objective of this test plan is to ensure a high level of confidence in the correctness and usefulness of the “Evaluation” and “Teamwork” modules of the TASS.

## Testing Strategy

The strategy for testing the “Evaluation” and “Teamwork” modules of the TASS is a combination of automated and manual tests. The components of the system that don't involve user interaction will be tested automatically. The components requiring tester interaction will be tested manually.

We will use Microsoft Test Manager to design the test cases.

## Scope

The scope of the current project is limited to the “Evaluation” and “Teamwork” modules of the student system and to the provided requirements. Testing such a system is a very “dependent” activity and this is why test plans and testing will be done by other teams to ensure its overall usefulness and correctness.

## Reference Material

* [Evaluation and Teamwork modules requirements (ETWMR)](https://github.com/QATeamApple/Documents/blob/master/Requirements/Requirements%20for%20Evaluation%20&%20Teamwork%20Modules%20v4.0.docx?raw=true)
* [Performance Test Plan (PTP)](https://github.com/QATeamApple/Documents/blob/master/Test%20Plan/Performance%20Test%20Plan/Performance%20Test%20Plan%20v1.0.docx?raw=true)

# FEATURES TO BE TESTED

The following is a list of the areas to be focused on during testing of the application.

## Admin UI

### Evaluation modules

* Practical Exams
* Practical Exam Files
* Practical Exam Evaluations
* Test Exams
* Students Records
* Export Results
* Bonuses
* Homework Evaluation
* Helpers
* Homework Statistics
* Homework Evaluation Reactions
* Add Students for Practical Exam
* Add Students for Test Exam

### Teamwork modules

* Teamwork
* Team Member Suggestions
* Import Teams
* Team Name Suggestions
* Import Team Names
* Team Member Evaluations
* Team Member All Evaluations

## Front-end UI

* Homework (Teamwork) Upload
* Homework Evaluation
* Homework Comments & Grading
* Exam Sign-up
* Exam Upload
* Exam Results
* Teamwork Sign-up
* Teammate Grading

# FEATURES NOT TO BE TESTED

The modules that will not be tested are dependent on and are assigned to other teams. The next sections with all included modules in them will not be tested:

* Basic Modules
* Users
* Courses and Seminars
* Anti-Cheat
* Specialties and Certificates
* Software Academy
* Calendars
* Test System
* Exam Evaluation

However, since testing requires operations from other sections, the testing team will use some of the other functions of the system.

# TESTING APPROACH

The testing will cover the requirements and efficiently reduce the risks.

## Integration Testing

Integration tests will exercise the interaction of the “Evaluation” module with other modules that it depends on. The goal is to ensure that they work nicely together.

## System Testing

System testing will be handled through manual tests and automated tests.

### Functional Testing

The objective of this test is to ensure that each element of the application meets the functional requirements of the business as outlined in the ETWMR.

* **User Interface Testing** will ensure that the user interface (UI) provides the user with the appropriate access and navigation through the functions of the target-of-test. In addition, UI testing ensures that the objects within the function of the UI work as expected and conform to corporate or industry standards.
* This stage will also include **Validation Testing** which is intensive testing of the designed front-end fields - valid, invalid and limit data input.

### Non-functional Testing

* **Performance Testing** will be conducted automatically. It will be done by using black-box testing method. Testers will interact with the user interface to the system and determine whether or not the system responds in a reasonable time. Reasonable is defined as the amount of time a data entry person would expect the system to respond in. The tests will be performed on the most common used browsers (IE, Chrome and Firefox).

Separate document ([Performance Test Plan (PTP)](https://github.com/QATeamApple/Documents/blob/master/Test%20Plan/Performance%20Test%20Plan/Performance%20Test%20Plan%20v1.0.docx?raw=true)) is created with more detailed information about Performance Testing.

* **Load Testing** will test the behavior of the system when an increasing number of users work with it at the same time.
* **Stress Testing** will test system behavior when overloaded. This test is particularly important for exam uploading.

### Regression Testing

When a change is made to the system, all test cases for all components relating (directly or indirectly) to the modified component will be re-executed. The design is to execute all tests necessary to ensure no regression occurs but not to needlessly expend resources on unrelated tests.

## Acceptance Testing

This test, which is planned and executed by testers, ensures that the system operates in the manner expected. Testers will provide feedback regarding changes, which must be implemented to the functionality and the UI. Feedback will be provided in the form of verbal communication at meetings and via email reports as necessary. All major changes will be documented in revised versions of the test plan.

# MEASURES AND METRICS

* Test coverage
* UI – buttons, menus, dropdowns, forms, etc.
* Coverage percent of the required features
* Defects
  + Ratio between total found and total fixed bugs
* Test Cases & Bug Priority metrics
  + **1 (Immediate)** – Defects must be resolved as soon as possible, because the defects are affecting the system severely. System/Feature is unusable until the problem is fixed.
  + **2 (Next release)** – Defects should be resolved in the normal course of development activities. The problems are not as severe and can wait until the new build/version is created.
  + **3 (On occasion)** – Defects that have minor impact on the user or have many workarounds and should be resolved after more severe defects have been fixed.
  + **4 (Open)** – Defects that have very low visibility and don’t impact the functionality in any way. These should be resolved if there is time and resources and there are no other higher priority defects present.
* Bug Severity metrics
  + **1 (Critical)** – Defects that render the system/feature unusable with the possibility of data corruption and no acceptable workaround.
  + **2 (High)** – Defects do not result in system/feature termination, but cause the system to produce incorrect, incomplete or inconsistent results with possible workarounds.
  + **3 (Medium)** – Defects do not result in termination and do not damage the system/feature usability. The results can be easily obtained with minor workarounds.
  + **4 (Low)** - Defects that are related to cosmetic issues and have many workarounds and low visibility to users.

# TESTING CRITERIAS

## System Test Entrance / Exit Criteria

### Entrance criteria

* The testing environment described in the environment requirements must be set up and functioning properly.
* All the necessary documentation, design and requirements information should be available that will allow the testers assure the system’s correct behavior.
* All the personnel involved in the testing must have the needed training to use the tools for the testing.
* Prepare appropriate test data or database.
* The system needs to be fully functional in order to perform performance testing as any malfunction will distort the results or interrupt the testing process.

### Exit criteria

* Certain level of requirements is achieved.
* No high-priority or severe bugs are left outstanding.
* All high-risk areas have been fully tested.
* Performance testing results meet requirements.
* The schedule has been achieved.

## Pass / Fail Criteria

The entrance criteria's for each phase of testing must be met before the next phase can commence. Now the criteria’s for pass and fail are given below.

## Suspension / Resumption Criteria

### Suspension Criteria

Test case execution will be suspended if a critical failure that impedes the ability or value in performing the associated test(s) is discovered.

### Resumption Criteria

Test case execution will be resumed when the problem causing suspension has been fixed. All test cases that deal with the modified part of the project will be re-executed.

## Approval Criteria

* The results of each test case will be considered “approved” if the results meet the expected results description in the test case.
* The results of performance testing will be considered “approved” if they meet the expected requirements in the [Performance Test Plan (PTP)](https://github.com/QATeamApple/Documents/blob/master/Test%20Plan/Performance%20Test%20Plan/Performance%20Test%20Plan%20v1.0.docx?raw=true).

# TESTING PROCESS

## Test Deliverables

A test report will be included in the project deliverables. This report will contain the set of test cases, a history of all formal test executions and a summary of the final state of the test suite.

## Testing Tasks

* Develop Test Cases
* Develop scripts for the automated tasks
* Execute tests
* Report defects
* Complete test report
* Manage change

## Responsibilities

All testers are responsible for the completion of all testing tasks.

The members of the team are responsible for approving the Test Plan and the Test Cases. They are also responsible for making reviews of the demonstrations and final acceptance of all work products.

## Schedule

|  |  |  |
| --- | --- | --- |
| Task | Deliverable | Week Performed |
| Develop test cases | Test Cases document | 26.09.14 - 03.10.14 |
| Develop test cases Develop scripts for automated testing | Test Scripts | 03.10.14 - 10.10.14 |
| Develop test cases Develop scripts for automated testing Prepare testing database | Test Data | 10.10.14 - 17.10.14 |
| Develop test cases Develop scripts for automated testing Execute tests | Daily Test Reports | 17.10.14 - 24.10.14 |
| Develop test cases Develop scripts for automated testing Execute tests Report defects | Weekly Bug Reports | 24.10.14 - 30.10.14 |
| Develop test cases Develop scripts for automated testing Execute tests Report defects Complete test report | Test Case Report | 30.10.14 - 07.11.14 |
| Develop scripts for performance testing  Develop load and stress test scenarios Execute tests Report results | Performance testing and weekly report | 07.11.14 - 21.11.14 |
| Create reports for regression testing and final reports. Update test plans and make final adjustments. | Regression testing. Final reports. Final versions of test plans. | 21.11.14 - 25.11.14 |

# ENVIRONMENTAL REQUIREMENTS

## Hardware

* Intel (R) Xeon (R) CPU @ 3.1GHz (Quad Core)
* Ram 8GB
* HD 100GB

## Software

* Windows Server 2008 R2
* IIS7
* MS SQL Server 2008 R2

## Access

* Administrator access has to be provided.

## Tools

* Microsoft Test Manager 2013 – for test case management system



* Telerik Test Studio – initially used for automated testing (using IE 10)
* Telerik Testing Framework – used for automated testing (using IE 10)
* Telerik Team Pulse – used for logging bugs
* Visual Studio 2012-2013
* GitHub – source control system

# RISKS AND MITIGATION

|  |  |
| --- | --- |
| Mitigation | Risk |
| Good communication between testing teams. | Dependency on other modules of the [TASS](http://test.telerikacademy.com/) |
| Make a team meeting ASAP. Re-evaluate new conditions and update the Test plan. | Requirements change |
| Tests cannot be conducted, so we can work on other issues of the Test plan. Update schedule and re-evaluate if needed. | [TASS](http://test.telerikacademy.com/) is down |
| Distribute workload. | Team member is unable to do his/her work for health or personal reasons |

# PLAN APPROVALS

|  |  |  |
| --- | --- | --- |
| Signature | Name | Date |
|  | Kiril Todorov | 25/11/2014 |
|  | Mladen Mladenov | 25/11/2014 |
| sign.png | Stanislav Iliev | 25/11/2014 |
|  | Svetlin Nyagolov | 25/11/2014 |
|  | Valeria Dimitrova | 25/11/2014 |
|  | Vanina Nenova | 25/11/2014 |