Software Requirements Specification

for

College Quiz System

**Version 1.0.5 approved**

**Prepared by Chris Knue, Matthew Yan, Johann Saumers, Gabriel Neumann, Brandon Hare, Cameron Cooke**

**Group 1**

**February 16, 2016**

**Table of Contents**

**Table of Contents**

**Revision History**

**1.** **Introduction**

1.1 Purpose

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Product Scope

1.5 References

**2.** **Overall Description**

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

**3.** **External Interface Requirements**

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

**4.** **System Features**

4.1 Add Quizzes and Questions

4.2 Connect to Existing User System

4.3 Take Quizzes

4.4 Quizzes Automatically Graded

4.5 Question Pools

4.6 Recommended Questions

4.7 Case by Case Extensions

4.8 System Logs

**5.** **Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

5.5 Business Rules

**6.** **Other Requirements**

**Appendix A: Glossary**

**Appendix B: To Be Determined List**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Matthew Yan | 2/16/16 | Added section 3 | 1.0.0 |
| Johann Saumer | 2/16/16 | Added section 5 | 1.0.1 |
| Christopher Knue | 2/16/16 | Added section 4 | 1.0.2 |
| Gabriel Neumann | 2/16/16 | Added section 1 | 1.0.3 |
| Brandon Hare | 2/16/16 | Added section 2 | 1.0.4 |
| Cameron Cooke | 2/16/16 | Added section 6 | 1.0.5 |

# Introduction

## Purpose

The purpose of our product is to provide an easy and simple system to enable teacher to give out quizzes or tests to students. The main focus of our system is to ensure the process is effortless for all individuals involved. This Specification Requirements Specification outlines the entire Quiz System and functions.

## Document Conventions

Our requirements are explicitly outlined within this document. Each system feature is outlined and includes the functional requirements that it covers. They are also assessed a value of priority, benefit, penalty, cost, and risk all parts involved with their implementation and their involvement with the system. Non-functional requirements are outlined afterwards.

## Intended Audience and Reading Suggestions

This document should be read in sequential order as written. These specifications intended to inform the reader, who is any stakeholder of the product from developers to users, the overall explanation of our product and its context. Section 1 establishes the initial premise of our System which is later elaborated on in the Sections following. Section 3 goes into the specifics of expected external requirements before outlining the specifics of the quiz system. These specification are required for use with our system and need to be implemented prior to the quiz system. For features and requirements of the College Quiz System see Section 4, or Section 5 for specifically Non-Functional Requirements.

## Product Scope

Our implementation of a College Quiz System is to provide users with a working and feature complete network for creating, editing, grading, and taking quizzes. Teachers can create and edit quizzes, setup in a way of their choosing, for students. Students can take these created quizzes at the time specified by their teacher. These quizzes are then graded either by the teachers themselves or by other users, such as teacher’s assistants, who have the privilege to grade the quizzes. Information about the quiz and question results is available to the teacher for personal interpretation. This is to accomplish the task of easily provided professors a way to easily create, manage, and grade quizzes for students to simply take online.

# Overall Description

## Product Perspective

The college quiz system is a new system, designed to be integrated within an existing third-party Moodle system. The systems will securely communicate student information and resulting quiz grades between each other.

## Product Functions

* Instructors can create quizzes
* Instructors can add and remove questions from quizzes
* The system can integrate with Moodle
* Students can take quizzes during specified times
* Students can take quizzes
* Students can see whether they got a question right or wrong
* Questions shall be graded automatically by the system
* TAs and Instructors can override grades
* The system can select a set number of questions for each quiz
* Teachers can give extensions to a student
* All actions are logged

## User Classes and Characteristics

Listed in decreasing order of satisfaction importance:

Instructors and Department Heads

* The teachers of the college’s courses. These staff members will be able to assign, manage and grade quizzes, as well as evaluate students and a course as a whole.

Students

* Students of the college who will be taking the quizzes. They have the lowest privileges, only being allowed to take quizzes and see their own results.

TAs

* Teacher’s Assistants are lower-level employees (most likely students themselves) responsible for grading quizzes. They do not have sufficient privileges to create or manage quizzes.

IT Staff

* Responsible for installing and maintaining the system. IT staff will have the have the highest privileges, being able to register instructors and students with the system.

## Operating Environment

The software shall be integrated with the college’s existing Moodle system. It can be installed on a local server within the school, preferably wherever Moodle is installed, or an off-site server if necessary. The software is used via any modern web browser both within and outside the school network. As it is a web application, the software should be cross-platform and run on the school’s current server hardware. The software will integrate cleanly with Moodle but not disrupt it or any other systems in the school.

## Design and Implementation Constraints

The software is to be closely integrated with Moodle, and as such should match the architecture and design of the system closely to allow easier integration. The security and integrity of users’ data must be maintained, only being revealed to authorized parties. The software must run efficiently, not consuming more resources than it requires (hard disk space, memory use, and CPU time). The system should respond within five seconds of user input. The school’s IT department will be responsible for installing and maintaining the software, so these tasks should be relatively straightforward and very well documented.

## User Documentation

The software will be delivered with an installation guide, technical troubleshooting guide, and a comprehensive user manual. Most pages on the web app will self-document with descriptions of possible actions, help popups (when triggered), and examples.

## Assumptions and Dependencies

We assume the software will be integrated closely with an existing, established Moodle system, and installed on a local school server. Users will have different levels of permissions and will only be able to view authorized data. Some baseline software components shall be reused for basic functionality (e.g. network security and encryption, SQL databases).

# External Interface Requirements

**3.1 User Interfaces**

This software’s user interface will consist of several key objects: informational text, errors, question input fields, help, and buttons.

Informational text will be the driving force for most of the system. This text will be formatted as one of the following types: header, sub-header, body, and question. This format will be kept for the entire quiz system.

Errors will always appear in bold, red text in the top right of the screen. These errors will be displayed in a pop-up box, and will hold window focus, requiring that the user respond to the message before proceeding.

Question input fields will exist in two forms: multiple-choice, and free-input. Multiple choice questions will be listed in either a bullet list, or a list of checkboxes, depending on the question. Free-input questions will be displayed as a text box, matching ⅔ the width of the website window. All fields will be listed vertically, with the actual question written out above the input fields.

The help button will be available on each screen in the bottom-left corner of the window. Clicking on the help button will take the user to a help page, when a FAQ will be listed, as well as instructions for how to obtain additional help.

All buttons will match the operating system’s buttons. These buttons will be set to a lighter shade of their original color when they are moused over, and will be “depressed” when clicked on.

**3.2 Hardware Interfaces**

It is required that the users use either a laptop or desktop computer with an operating system that supports a HTML-capable web browser. This browser will connect to the quiz server at a specified web address, unique to each school. The user will use a USB-connected mouse to interact with the on-screen interface, as well as a USB-connected keyboard to input text into text fields.

**3.3 Software Interfaces**

This product will interface with the Moodle 2.8 course-management tool to input user grades into the system.This software will also use Moodle to gather a list of classes, as well as a list of enrolled students per-class. The quiz system will be driven by Adobe Flash 20, which will need to be installed on all applicable machines.

The school’s quiz database server (Server) will be contacted by the user’s machine (Client) through its respective web-browser. This initial message from the client will be a triple handshake that includes hashed login information, as well as the IP address for logging. The Server will use this information to look up class lists and currently open quizzes and send this information back to the Client.

The Client will be able to try and begin a quiz. This will send a message to the Server, sending the hashed quiz ID send in the original messages. The Server will validate the request, and upon success send the quiz questions to the client, so that they may begin to take the quiz.

Each question answered in the quiz will be sent to the Server in an encrypted form. The server will validate this request, and upon success will notify the Client that their answer was saved.

If at any point in the message sending a problem occurs, the Server will dispatch a message to the Client, notifying them of the problem, and the steps to take to correct the discrepancy.

**3.4 Communication Interfaces**

This product will communicate with external systems through an HTML web-browser. All messages from Client to Server, or vice versa, will be encrypted using MD6 and sent over HTTP. Due to the heavy cost of encrypting large amounts of data, open text fields will be sent as plain-text. Data transfer will be limited to a maximum speed of 1 MB/s, to help reduce the strain on the Server.

All messages will be validated with a triple handshake. When a message is received, the recipient will send a confirmation message back to the sender, who will respond in kind. If any of the messages are not received within a time-out period, the message as a whole will be discarded, and an error message will be dispatched back to the appropriate party.

# System Features

## Add Quizzes and Questions

### Description and Priority

Teachers can add quizzes for students to take. Teachers can also add questions to quizzes.

#### Priority: High

The software cannot ship without this feature.

#### Benefit: 9

This is the core feature of this software

#### Penalty: 1

#### Cost: 5

This feature will be the core focus of development

#### Risk: 8

This Failure to have this feature will result in a non-shippable state

### Stimulus/Response Sequences

* Teacher presses create quiz button
  + System creates quiz
    - Teacher presses create question button
      * System adds question to quiz

### Functional Requirements

REQ-1: Quizzes shall be able to have a name set by the instructor

REQ-2: Quizzes shall be able to have a topic set by the instructor

REQ-3: Quizzes shall be able to have a time limit set by the instructor

REQ-4: Quizzes shall be able to be assigned to a course

REQ-5: Quiz Questions shall be either right or wrong

REQ-6: Quizzes shall be able to have ranges of time they are available during set by instructor

REQ-7: Instructors shall be able to change anything about the quiz after it is made

## Connect to existing user system

### Description and Priority

The software is capable of connecting to existing user systems (such as Moodle) to provide easier integration of this system into the College ecosystem.

#### Priority: High

#### Benefit: 7

This feature shall cut the cost of development by reducing the amount of work we need to do. It will also cut costs for colleges because data about faculty and students will not have to be re-entered into a new system.

#### Penalty: 2

This feature removes some control over what information is available

#### Cost: 1

This feature should reduce overall costs

#### Risk: 2

### Stimulus/Response Sequences

* Admin enters credential to access existing user system
* System access existing user system

### Functional Requirements

REQ-1: Admins shall be able to connect to existing user systems

## Take quizzes

### Description and Priority

Students can take quizzes that were created by teachers during the times specified.

#### Priority: High

The software cannot ship without this feature

#### Benefit: 9

This feature is a core part of the software

#### Penalty: 1

#### Cost: 5

This is a core feature, much of the time of development will be here

#### Risk: 8

Not having this feature means not shipping

### Stimulus/Response Sequences

* Student attempts to access quiz during quiz time
* System gives student quiz
  + Student completes quiz
    - System shows student what questions they got right
* Student attempts to access quiz outside of quiz time
* System denies student access to quiz

### Functional Requirements

REQ-1: Students shall be able to take quizzes

REQ-2: Students shall be able to see whether they got a question right or wrong once quiz is completed

## Quizzes automatically graded

### Description and Priority

Quizzes will grade all questions that instructors have not marked as “Manual Grade”

#### Priority: Medium

#### Benefit: 6

This feature makes our quiz system easier than paper tests

#### Penalty: 1

#### Cost: 3

#### Risk: 3

### Stimulus/Response Sequences

* Student completes test
  + System grades all auto grade questions

### Functional Requirements

REQ-1: Questions shall be able to be graded automatically by the system

REQ-2: TAs/Instructors shall be able to override grades given by the system

## Question pools

### Description and Priority

Quizzes will can have more questions available than are actually used in the quiz and can shuffle them in randomly per “section” of quiz

#### Priority: Low

#### Benefit: 4

This feature makes our quiz system more appealing than paper tests by making it more difficult to copy answers from friends

#### Penalty: 1

Increased complexity

#### Cost: 3

#### Risk: 2

### Stimulus/Response Sequences

* Teacher adds question to question pool
  + Student access quiz
    - System generates quiz from pool
      * System gives student quiz

### Functional Requirements

REQ-1: Questions shall be able to be forced into every quiz

REQ-2: The system shall be able to select a set number of questions for each quiz

## Recommended questions

### Description and Priority

When making a quiz, teachers will be prompted to add questions from the most resent quiz that many students failed.

#### Priority: Low

#### Benefit: 5

This feature makes our quiz system easier than paper tests

#### Penalty: 1

#### Cost: 5

#### Risk: 3

### Stimulus/Response Sequences

* Teacher creates quiz
* System asks teacher if they want questions from previous quiz with high percentage of failure
  + Teacher selects yes
    - Questions are added
  + Teacher selects no
  + Teacher selects never
    - System stops asking teacher

### Functional Requirements

REQ-1: Teachers shall be able to disable this prompt for themselves

## Case by case extensions

### Description and Priority

Teachers can give extensions to individual students for taking a quiz

#### Priority: Medium

#### Benefit: 4

#### Penalty: 1

#### Cost: 3

#### Risk: 3

### Stimulus/Response Sequences

* Teacher gives student extension
* System allows student to access quiz until the new due date

### Functional Requirements

REQ-1: Teachers shall be able to extend the due date for individual students

## System logs

### Description and Priority

All actions are logged for Admins to understand what is going on

#### Priority: Low

#### Benefit: 6

This feature should make it easier to debug

#### Penalty: 3

This could require immense disk space and potentially slow the software

#### Cost: 4

#### Risk: 2

### Stimulus/Response Sequences

* User performs action
* System logs action to file
* Admin disables logging
* System stops logging after logging that it is to stop logging

### Functional Requirements

REQ-1: Admins shall be able to turn of logging

# Other Nonfunctional Requirements

## Performance Requirements

The main performance requirement is to make sure the system responds at a suitable rate. It needs to respond fast enough so that if something happens to the connection to the internet on the user-side, they do not lose a lot of progress when they resume. There should be approximately a maximum of 5 seconds of response time to input to ensure that the system is still running. After this time they should be prompted to check their connection and try again.

## Safety Requirements

As this product is a browser based application, there are minimal threats to physical damage. The most damage that could result is from theft of private information, which is covered in the following section.

## Security Requirements

All logins must be validated through username and password. The username/password system is an external system that will be handled outside of our product. The product simply prompts the user for a username and password and validates that the provided information is correct with the external system.

## Software Quality Attributes

The system must be very easy to use. This is the most important attribute. It should be easy for teachers to create a quiz and for students to take the quiz. It should be usable by people with minimal computer knowledge.

It should be flexible to handle connections to external systems. The external systems can include usernames and passwords (SRS), grade keeping (Moodle), etc.

It must also be reliable. It should save the quiz periodically on creation so the teacher loses minimal work in the case of an internet connection error. The quiz answers should also be saved if the student should happen to lose connection. This way the system will not only be reliable and not crash often, but in the event of a crash there will be minimal loss of data.

## Business Rules

Teachers are the only ones able to create and edit quizzes. The students are only able to take the quizzes. Answers can’t be edited after submission by either party to ensure that teachers won’t be able to aid or spite any student.

# Other Requirements

The system needs to adhere to all local laws in order to be legal. The user of the system will be required to have a database in order to store quizzes and results for each course that uses the system, along with the appropriate logs.

**Appendix A: Glossary**

**-Acronyms**

* FAQ: frequently asked questions
* REQ: requirement
* HTML: Hypertext Mark-up Language
* HTTP: Hypertext Transfer Protocol
* MD6: cryptographic hash algorithm
* TA: teacher assistant
* IT: information technology
* CPU: Central processing unit
* SQL: structured query language

**Appendix B: To Be Determined List**

* The ability to print out the quiz and complete it by hand, then scan and submit the quiz to be graded.
* Support the ability to write in different languages.