

**SYSTEM OVERVIEW REPORT - VERSION**

**WEB BASED PROGRAM Q&A AND INFORMATION SYSTEM**

|  |  |
| --- | --- |
| **Prepared by:** |  |
| Consulting Team Name: | **TeeFour** |
| Project Team Leader: | Chang Xing Li |
| Team Members: | Keith Sandy  Gurbachan Singh  Paul Trudel  Juan Antonio Villamizar  Matthew Zigman |

***<2016-4-13>***

# 

# Acknowledgements

**The Client**

Linda Crane

**The Tee-Four Technologies Team**

Chang Xing (Calvin) Li - Team Leader

Keith Sandy - Team Member

Gurbachan Singh - Team Member

Paul Trudel - Team Member

Juan Antonio Villamizar - Team Member

Matthew Zigman - Team Member

**The Project Professor**

Patricia Murphy - Algonquin College

**The Project Advisor**

Todd Kelley - Algonquin College

**Table of Contents**

Acknowledgements i

Executive Summary i

System Overview 2

Data Design 3

Architectural Design 4

Program Design 6

Conclusion 14

Recommendations 14

References 15

Appendices 16

**List of Tables and Figures**

Table 1: Cross Reference Table 6

Figure 1: ERD Diagram of Program Selection System 4

Figure 2: Physical Architecture of the Program Selection Selection 5

Figure 3: Use Case Diagram 13

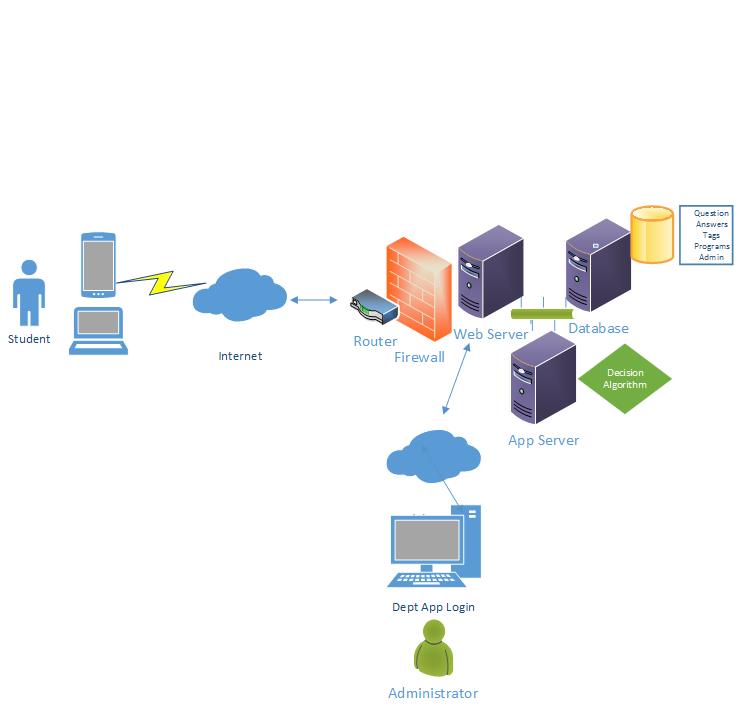
Figure 4: Sequence Diagram 13

# System Overview

The system overview (1-2 pages) will show the purpose of the system, architectural design, the major features of the system and **security**.

The proposed student program selection system physical architecture comprise of a web-based frontend access by students and a server-based application administrative backend deployed on an existing wordpress content management system of the college. The diagram below depicts the physical architecture of the program selection system.

Figure 2: Physical Architecture of the Program Selection Selection



When the program is up and running the students using their mobile phones or laptops will click on a special button at the department’s program main web page to start the process of getting help in selecting a suitable IT program. The students will be asked to respond to a series of questions and based on their answers that will be stored in the database, the application will automatically recommends the most suitable programs.

The department administrator will have a login access to the application on the existing wordpress dashboard. The administrator is required to upload a series of questions, tags and all the available programs. The PHP application will store the information in the existing wordpress database with a additional tables catered for the application. The application will include a ‘smart’ decision-making algorithm that will use the answers and tags to determine the most appropriate program for the student. The result will be automatically displayed to the student immediately after he has completed all the questions.

The overall application solution will involve programming with with PHP, JQuery, Javascript and CSS, WordPress CMS using existing web server and database resources. The project will adopt an agile adaptive development strategy where there will be regular engagements with the client and a series of sprint events to iteratively ‘facilitates and embraces changes’ (Subrahmanya & Lakshmanan, 2012).

Security

TeeFour has taken many measures in order to ensure the security of this system. The first step in making any system secure is identifying areas of a system which may be vulnerable to an attack and to which type of attack. The areas we found to pose potential problems were all on the administrative side of the application. The types of attacks we identified as being a potential danger to the integrity of this system were SQL injection and XSS. In order to combat these attacks we sanitize all user input before it enters the database through the use of built in PHP functions.

Another area that we thought might be of concern was admin privileges. In compliance with the concept of least privilege we have made it so only the root administrator has complete control over the system. All other administrators are delegated privileges by the root administrator and are restricted to areas of the system that they have been given permission to.

The last security measure taken is in regards to password security. It is considered bad practice and insecure to store plaintext passwords in a database. In order to overcome this security risk we have employed the use of a hashing algorithm in order to ensure the security of our passwords. It is important that the algorithm chosen is strong enough and as such multiple algorithms were taken into consideration. MD5 and SHA128 were the first algorithms that were looked into but as of now they are both insufficient in stopping the retrieval of passwords. The hashing algorithm sha256 was decided upon and has been implemented into the system in order to ensure password security.

# Hardware and Software Requirements

Identify recommended hardware specifications and software environment (including operating system, browser, …).

The recommended hardware is:

* 1 gigahertz (GHz) or faster 32-bit (x86) or 64-bit (x64) processor\*
* 1 gigabyte (GB) RAM (32-bit) or 2 GB RAM (64-bit)
* 25 GB available hard disk space (32-bit) or 30 GB (64-bit)
* DirectX 9 graphics device with WDDM 1.0 or higher driver

It is recommended that the user uses Windows Operating System running either Google Chrome or Firefox Browser.

# Installation

Running The System On A Local Host/Local Machine

Step 1

If you wish to run this system locally on your machine, then you will need a cross platform webserver solution that will provide a webserver and SQL server. You can download this application at<https://www.apachefriends.org/index.html>. Click the download link in accordance to your type of operating system. Begin the installation and be sure to keep everything at its default settings, its important you do not change any preferences or directories. After the installation has completed please open the XAMPP control panel and press start on the Apache and MySQL module. If you have any problems starting the modules, please refer to the text log and adjust port settings for any application that may be listening on the ports Apache and MySQL are trying to listen on.

Step 2

Now you will have to initiate a database using phpmyadmin. Make sure you have your Apache and MySQL servers running then put http://localhost/phpmyadmin/ into your default browsers address field. Once you are on the page then proceed to go to the import tab at the top. Click the “Import” button and within the projects folder go to sql folder, and select the pss.sql file. Once you have selected this file, click the go button. It will take a minute so please be patient because if you click go again while its executing then it might cause errors.

Step 3

Now you will need to move the project folder into a directory where it can be accessible by XAMPP. The directory you will need to put the project in is at C:\xampp\htdocs.

Step 4

Once the project in the htdocs folder and the Apache and MySQL servers are running, you can now access it locally. Open your default browser and put<http://localhost/pss/viewlayer/admin/>

For the administration side and put

<http://localhost/pss/viewlayer/student/>

for the student side.

Using the System

Administrators

Administrators have the ability to make changes to this system. There are two types of administrators, a regular admin and the root admin. The root admin is added during database creation. The root admin controls the entire system is the only user which can create other users and can edit all parts of the system. A regular user is only able to access areas of the program to which they have been given privileges to.

Departments

All users have the ability to add a new department as well as add programs to that department. When a department is created only two users have access to it by default; the user who created the department and the root admin. Once the department is created the root admin can add users to the department and remove them.

Tags

Tags are keywords which are associated with both programs and answers. Tags describe an aspect of the program to which they belong to. Examples of tags would things like “Java”, “2 years”, and “Software”.

Questions and Answers

Questions are associated with departments and the answers to questions are associated to tags. Questions can belong to more than one department and answers can belong to more than one tag. To edit an already existing question a user must belong to at least one of the associated departments of the question.

To create a question you must provide three pieces of information: the question text, the weight of the question, and the associated department(s).

Example.

Question Text: How long would you like to go to school?

Weight: 2

Departments: All

The weight of a question will affect the final score for the user when they complete the test. The weight acts as a multiplier for the score of the question therefore a weight of two will double the value of the question and a weight of zero will nullify the question.

Once you have added the question you will be redirected to a page to add the answers to the question. Remember answers have tags associated with them so you should create the tags that you wish to use beforehand.

Example.

Question: How long would you like to go to school?

Answer 1: 1 year, Tag: 1 year

Answer 2: 2 years, Tag: 2 years

Answer 3: 3 years, Tag: 3 years

When you are satisfied with your answers click the “Submit Answer” button to add it to the question. The button will then take you to a new page to add another question. Continue this process until you have added all the questions you want and then click the “Done” button (the “Done” button does not add the answer you are currently on to the question). It is important to note that an answer can have be associated with any number of tags that you wish including none. You would make use of no tags for an answer like “none of the above”.

You can return to the question you have created another time and change its text, weight, and departments. You can also add, remove, and change the order of answers for the question.

The Completed Test

You can view the compiled test for each department by selecting it from the “Questions” sub menu. From this screen you have the ability to change the order of the questions the user will see when they take the test.

Notes

This section will cover some important things to take into consideration when attempting to implement this system.

The first thing that should be noted is how a student will come to see the test. Each department will possess their own unique test. This test is accessed when the user clicks a button on either the department’s page or on the page of a program associated with the department. In order to serve the user the proper test the webpage must have a unique identifier for the department embedded into it. As of now this unique identifier is the department’s code.

This unique identifier allows the program to find the department in the database and then find the proper test which is associated with that department. The program will not function properly without this.

Another thing to note is the location of the algorithm which scores the test and gives the user their results. The algorithm can be found in the “CalculateResult.php” file in the “dataaccesslayer” directory.

# Data Design

The data design for this Questionnaire will implement the standards used for a web-based survey. Where at the back end data will be inserted, updated and manipulated by SQL through the use of PHP in order to be displayed in a web environment.

MySQL integrated with PHP will be used for the development and production of this questionnaire. Tables will be created according to questionnaires and users will be created to grant access to the existing tables and also to enable the user to create questionnaires that will use fixed tables for specific parameters that pertain to the questionnaire structure.

Requirements are available since the Client’s server uses Tomcat thus making it possible for the team to integrate the back-end MySQL with the website. Data space has to be discussed with the client.

For the ERD diagram we have created a set of tables for department, questions, tags and program. In order to avoid possible many-to-many relationships, we have created linking tables for department-admin (DALinks), question-program(QPLinks), tag-program links (TPLinks) and finally, tag-question links(TQLinks).

# Program Design

This section shall describe the functionality that will be present within our system upon its completion. The languages which will be used to achieve these requirements are PHP, SQL, HTML, and Javascript. During development and testing we will be monitoring how information is sent into the program and the database. We will do this to make sure that all information is validated and cleaned as to protect from potential SQL injection, XSS, and other web based attacks.

1. Cross Reference Table - list functional requirements and identify associated use case and sequence diagrams.

**Priority scale:**  1 (optional) - 5 (completely necessary)

|  |  |
| --- | --- |
| **SWF1** | **Administrator Creation and Deletion** |
| **Functional Requirement**  **UC1**  **S1** | The system shall allow administrators which have been given additional privileges to add or remove new administrators. |
| **Description** | Administrators which have been granted special privileges from the program's super user(s) will been given the option to create a new admin. This new admin will be added to one or more departments with the ability to modify aspects of that department which relate to the rest of the system. Tying into this admins may be added to more departments after creation  Administrator which have been granted special privileges from the program’s super user(s) will be given the option to remove admins from the system. When an admin has been removed from the system they will no longer have to ability to login or make anymore changes to the program. Admins which have been assigned to more than one department may be removed from one department thereby losing all privileges relating to that department but will still retain their administrative privileges to all other departments to which they have been assigned. When an admin is completely removed from the system or simply removed from a department any changes which they have made will remain intact. |
| **Priority** | 5 |

|  |  |
| --- | --- |
| **SWF2** | **Tag Creation and Deletion** |
| **Functional Requirement**  **UC2**  **S1** | The system shall allow the creation and deletion of association tags |
| **Description** | Administrators shall have the option to add a tag. Administrators shall provide text into a field that will be saved to a database when the Administrator commits their changes by clicking the add button. Tags will be used later on in the question and answer creation phase to associate questions with their related program.  Administrators shall have the option to delete a tag. Administrators shall be able to view what tags are associated with what programs and remove them by selecting them and clicking the remove button. |
| **Priority** | 4 |

|  |  |
| --- | --- |
| **SWF3** | **Question Creation and Deletion** |
| **Functional Requirement**  **UC3**  **S1** | The system shall allow administrators to add and remove questions to and from the question bank. |
| **Description** | Administrators shall have the option to add new questions and provide answers. Administrators will be provided with text fields, where they can enter questions and choose what program they are related to. After the administrator provides their new question they will have to provide potential answers in text fields directly below the question field. The administrator can then save their questions to a database by clicking the add button.  Administrators shall have the option to delete current questions. All programs within a department will be displayed after the login screen. Administrators can select any program and have the questions pertaining to that program displayed. The administrator shall have the option to simply select any question and delete it by clicking the remove button. |
| **Priority** | 4 |

|  |  |
| --- | --- |
| **SWF4** | **Find Programs from answers** |
| **Functional Requirement** | The system shall be able to find programs based on the answers provided by the user. |
| **Description** | After the administrator provides their new question and answers, they will choose a tag which will relate a question to a specific program. The tag bridges the questions and their answers to their associated programs. This allows the system to gather the results from the user and provide them with a suitable program based on their answers to the question. |
| **Priority** | 5 |

|  |  |
| --- | --- |
| **SWF5** | **Gather questions based on program** |
| **Functional Requirement** | The system shall automatically gather questions based on their linked programs. |
| **Description** | If an administrator wishes to view any questions, they will have to view them by their related programs. The system will be able to determine what department each administrator resides in, after they logged in. Once they have logged in they will be presented with all programs within their department. By clicking on any of the programs they will be able to view all the questions related to that program. |
| **Priority** | 3 |

2) The Use Cases used in the analysis and design are listed as follows:

UC1

**Name:** Creation and Deletion of Administrators

**Actor:** Administrator

**Type:** Create and Delete Administrators

**Description**:

* + 1. Administrators which have been granted special privileges from the program's super user(s) will been given the option to create a new admin.
    2. This new admin will be added to one or more departments with the ability to modify aspects of that department which relate to the rest of the system.
    3. Tying into this admins may be added to more departments after creation
    4. Administrator which have been granted special privileges from the program’s super user(s) will be given the option to remove admins from the system.
    5. When an admin has been removed from the system they will no longer have to ability to login or make anymore changes to the program.
    6. Admins which have been assigned to more than one department may be removed from one department thereby losing all privileges relating to that department but will still retain their administrative privileges to all other departments to which they have been assigned.
    7. When an admin is completely removed from the system or simply removed from a department any changes which they have made will remain intact.

UC2

**Name:** Tag Creation and Deletion

**Actor:** Administrator

**Type:** Create and Delete Tags

**Description**:

* + 1. Administrators shall have the option to add a tag.
    2. Administrators shall provide text into a field that will be saved to a database when the Administrator commits their changes by clicking the add button.
    3. Tags will be used later on in the question and answer creation phase to associate questions with their related program.
    4. Administrators shall have the option to delete a tag.
    5. Administrators shall be able to view what tags are associated with what programs and remove them by selecting them and clicking the remove button.

UC3

**Name**: Question Creation and Deletion

**Actor**: Administrator

**Type**: Create and Delete Questions

**Description:**

i) Administrators shall have the option to add a question/

ii) Administrators shall write a questions which will be associated both with a program and with a set of answers.

iii) Questions will be used to determine a suitable program of study for the users of this system.

iv) Administrators shall have the option to delete a question

v) Administrators shall be able to remove questions from the database which will also remove all answers which were associated with that question.

UC4

**Name:** Find Program form Answers

**Actor:** Administrator

**Type:** Find Program form Answers

**Description**:

* + 1. After the administrator provides their new question and answers, they will choose a tag which will relate a question to a specific program.
    2. The tag bridges the questions and their answers to their associated programs.
    3. This allows the system to gather the results from the user and provide them with a suitable program based on their answers to the question.

UC5

**Name:** Questions Based on Program

**Actor:** Administrator

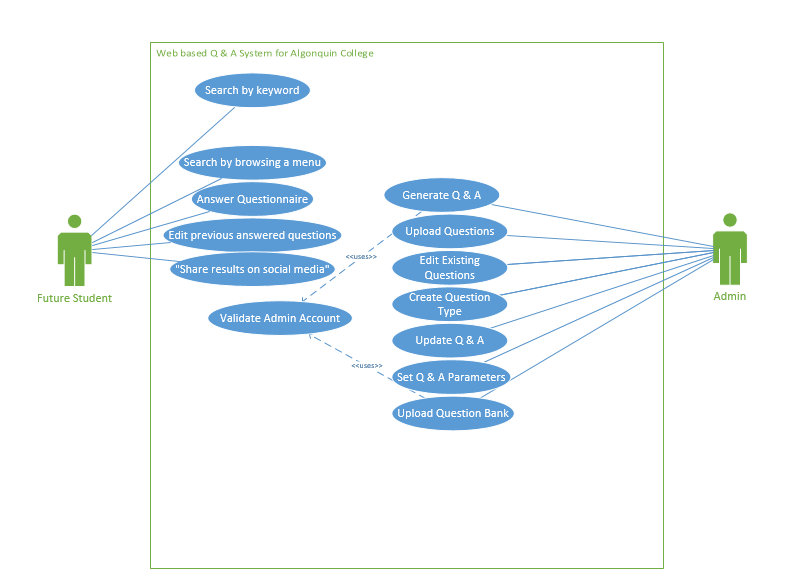
**Type:** Gather Questions based on Program

**Description**:

* + 1. If an administrator wishes to view any questions, they will have to view them by their related programs.
    2. The system will be able to determine what department each administrator resides in, after they logged in.
    3. Once they have logged in they will be presented with all programs within their department.
    4. By clicking on any of the programs they will be able to view all the questions related to that program.

The Use Case Diagram below summarises Administrator and Future Student actions performed within the boundary of the system.

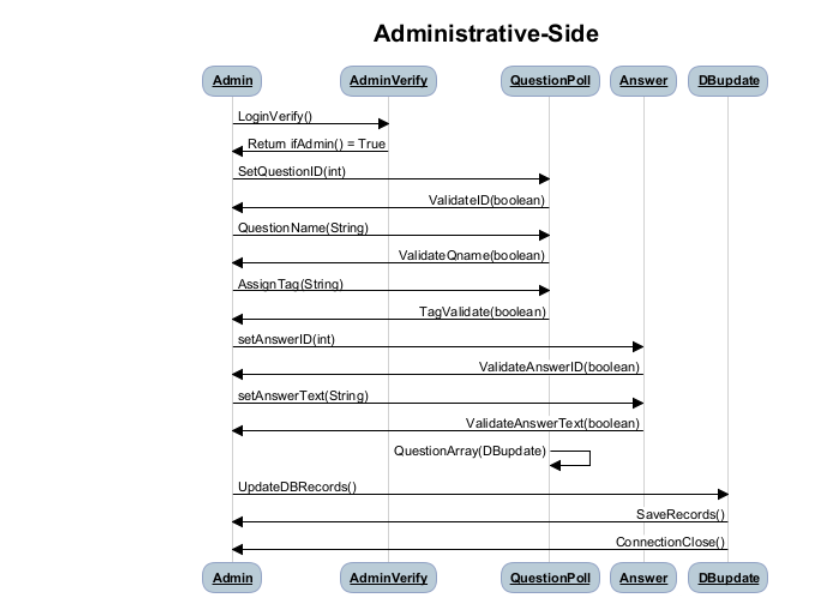
Figure 3: Use Case Diagram



3) Behavioural descriptions:

The sequence diagram is a sample example describing the general overview of the process behind what an administrator would do, the options are limited enough to have this diagram display what process is used. The administrator logs in and is authenticated, then they are prompted with adding question to pool (Removing too) or the option to quit the application

Figure 4: Sequence Diagram



|  |  |
| --- | --- |
| Rubrics: |  |
| Unsatisfactory | -does not meet minimal requirements |
| Developing | - first draft - meets minimal requirements |
| Ideas | - all required sections of template included as requested. |
| Connections | - attention to detail, integration of report sections, connected to client's needs |
| Extensions | - value-added content which defines the team as unique |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project Group Name:** | |  |  |  | **Normalized** |  |
| **System Overview** | | | **100.00** | **%** | **2.00** | **%** |
|  |  |  |  |  |  |  |
|  | Unsatisfactory | Developing | Ideas | Connections | Extensions |  |
|  | 0 - 49 Marks | 50 - 69 Marks | 70-79 Marks | 80-89 Marks | 90-100 Marks |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **1. System Overview (arch.,features,security)** | | | | **20.00** | /20 |  |
| **2. H/W & S/W** | | | | **20.00** | /20 |  |
| **3. Installation** | |  |  | **20.00** | /20 |  |
| **4. Data Design** | | | | **10.00** | /20 |  |
| **5. Program Design** | | | | **20.00** | /20 |  |
|  |  |  | **Total** | **100.00** | /100 |  |
| **Deductions (missing Source code table)** | | | | **0.00** | /10 |  |
| **Deductions (format/spelling/grammar)** | | | | **0.00** | /20 |  |
| **Total Marks** | |  |  | **100.00** | /100 |  |
|  |  |  |  |  |  |  |
| **Normalized Total** | | |  | **2.00** | /2 |  |