

AutoEd 2.0

An Online Courseware and Motivation System

by

Eric Wein

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF SCIENCE HONOURS

in

The Irving K. Barber School of Arts and Sciences
(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Okanagan)

April 2013

© Eric Wein 2013

Abstract

AutoEd 2.0 is a web based courseware system with an integrated user profile system that allows students to complete assignments and tests online. The system was developed to be a remodeling of a previous system (AutoEd [2]) to include a better user interface along with a points and badge system to keep students motivated. Each student upon registration is given a profile page where they can set goals, rate courses, view recommended courses and earn various badges. Students are also awarded points for their achievements whether it be obtaining a new badge or achieving an A on a test. As students earn points they go up in level to further award the more motivated students. The system was designed using various methods to keep the system modularized and easy to expand and build upon. This paper will demonstrate the use of gamification to increase student motivation in online courses and indicate whether or not students do better when using the AutoEd 2.0 system.

Table of Contents

Abstract	ii
Table of Figures.....	v
Acknowledgements.....	vi
1 Introduction.....	1
1.1 Motivation.....	1
1.2 Thesis Statement	2
2 Background.....	2
3 Motivation System	3
3.1 Approach.....	3
3.2 Points & Level System	4
3.3 Badges.....	5
3.4 Student Goals	6
3.5 Email Reminders.....	7
4 High Level Architecture.....	7
4.1 Modules.....	8
4.1.1 Test Completion	8
4.1.2 Profile System.....	9
4.2 System Overview.....	11
5 User Manual.....	12
5.1 Profile System	12
5.1.1 Logging In.....	12
5.1.2 Registration	13
5.1.3 Password Recovery.....	14

5.1.4 Student Profile Page	15
5.1.5 Transcript	16
5.1.6 Rate Content.....	16
5.1.7 Badges.....	18
5.1.8 Settings	19
5.1.9 System Header	20
5.2 Courseware System.....	20
5.2.1 Test Listings	20
5.2.2 Active Test	21
5.2.3 Past Due Test	21
6 Development.....	22
6.1 Challenges	22
7 Results and Feedback	23
8 Conclusions.....	23
9 Future Work	24
Appendix.....	26
References.....	41

Table of Figures

Figure 3.3: A screenshot of all badges available to the user	6
Figure 4: System architecture diagram	8
Figure 4.1.1: A screenshot of a sample question.....	9
Figure 4.1.2: A screenshot displaying a sample student profile page.....	10
Figure 4.2: Diagram showing data movement between the two systems.....	12
Figure 5.1.1: A screenshot showing the system login page	13
Figure 5.1.2: A screenshot showing the system registration page.....	14
Figure 5.1.3: A screenshot showing the student password recovery page.....	14
Figure 5.1.5: A screenshot showing the system transcript page.....	16
Figure 5.1.6: A screenshot showing the content rating page	17
Figure 5.1.7: A screenshot showing the student badges page.....	18
Figure 5.1.8: A screenshot showing the system settings page	19
Figure 5.1.9: A screenshot showing the system header.....	19
Figure 5.2.1: A screenshot showing the test listings for PHYS 122.....	20
Figure 5.2.3: A screenshot showing a sample past due test.....	22
Figure A1: A screenshot of the database DDL script	28

Acknowledgements

I would like to acknowledge Dr. Ramon Lawrence, my supervisor, for his guidance throughout this project. Without his continuous encouragement and supervision, the completion of this project would have been much more difficult.

I also thank Giuseppe Burtini who shared his extensive knowledge, expert opinions and development skills in the creation of this project.

Finally I would like to thank Alyosha Pushak for developing the original AutoEd [2] system, for without its key features this project would not have been completed.

1 Introduction

1.1 Motivation

While many online education systems exist today, they lack many of the features needed to keep students motivated. Most systems are strictly a question and answer approach which can be very dull for the student causing a lack of motivation and an increase in procrastination. When students lack motivation they leave their work until the last possible moment resulting in rushed work and poor results. Take Aplia [1] for example, this system is a great interactive online learning tool however the system does not give anything back to the student to keep them engaged and wanting to “play” longer on the system. By incorporating concepts such as user awards, goals and profile pages, the system could engage the student by portraying the feeling they are achieving something more than just completing a test [8].

Today’s youth are very familiar with gaming and the many aspects of gaming. By using these gaming concepts that students are already accustomed in a non-gaming environment we can create a familiar and relaxed atmosphere for the student. Furthermore, by using concepts that the student is already familiar with, the system learning curve is minimal allowing the student to use the system straight away to its full potential. Therefore, through the use of gaming concepts and cleaner user interfaces we can increase student engagement and motivation in online education systems [8]. To achieve this, a motivation system was built on top of an online courseware system to keep students engaged and increase student motivation to complete assignments.

1.2 Thesis Statement

In 2011, undergraduate student Alyosha Pushak developed the first courseware system known as AutoEd [2]. This system was developed into two parts, one for students and one for professors. Professors create question templates which are used to randomly generate questions making up a test. Students are assigned tests where they submit answers online and receive immediate feedback whether they have correctly answered the question. Students are also presented with a hint if they incorrectly answer the question to help aid them in their next attempt. Although the original AutoEd system contained many great features, its user interface was very dull and confusing and most importantly it was not engaging for the user.

The contribution of this thesis is the development of a new user interface including a motivation system for the AutoEd courseware system. This user interface includes a social profile system where users earn badges and points for their accomplishments as well as the ability to set goals and rate courses. The goal is to create an online courseware system to allow students to complete courses online while keeping them engaged and motivated to increase their performance in the course.

2 Background

The idea of an online system to complete assignments and tests is not new. There are many systems today such as Aplia [1] and Gradianc [3] that allow for online interactive learning. However these systems lack the features needed to keep students motivated. Aplia [1] for example is affiliated with many textbook publishers that allow Aplia [1] to be an interactive learning tool in conjunction with these textbooks. Although Aplia [1] has many great

features such as topic tutorials and intuitive test completion capabilities it is missing key features to keep students engaged and motivated.

A relatively new concept in the realm of student motivation is the idea of gamification. Gamification is the approach of using gaming concepts in non-gaming environments to make them more engaging and fun to use [7]. Completing tests and assignments online can be very stressful and unpleasant which leads to reduced student engagement and motivation. By using concepts of gaming which most students are already accustomed, one can make a normally stressful environment into a relaxed and fun environment. By offering badges, points and the ability to unlock achievements or gain precedence through the use of a level system a sense of incremental achievement can be portrayed to the student [8]. As students obtain these small gains through the completion of badges or by reaching a new level the sense of achievement is constant. This constant feel of achievement can boost student confidence, motivation and engagement in online learning.

3 Motivation System

3.1 Approach

Games are widely used and known for their immense entertainment capabilities. They bring out the competitive nature in humans which strives them to win the game at hand. Our approach to motivation was that of gamification which means taking concepts of gaming and using them in a non-gaming environment to increase student engagement and motivation [7]. By using these concepts we can create a sense of incremental achievement and make a competitive and challenging environment for the students which

in turn will increase student engagement and motivation in the AutoEd 2.0 system.

3.2 Points & Level System

In almost every video game imaginable there is the capability to level or rank up. With each new level gained, a player is given a more prestigious rank among the other players, resulting in a boost in confidence and overall enjoyment of the activity.

In the AutoEd 2.0 system, students are awarded points for their achievements, such as completing a new badge or completing a test. The better you do on a test the more points the student receives. For example, if a student completes a test with 100% that student is awarded 10 points where a student who completed the same test with 80% would receive 8 points. As students gain points they are also awarded with an increase in level. The student must acquire 100 points to move up in level. However as you go up in level it becomes more difficult to receive points. For example if a student has reached level 3 by completing all the available badges, the student can now only rely on their test scores to receive the same quantity of points. On each student profile page the students' level is displayed for themselves to see as well as other students to see when the students profile page is viewed. Along with the student's level, a progression bar is displayed showing how far the student has come and how far they have remaining to go to reach the next level. By adding in the level and points system the student is able to view their incremental progression. The points system is also used to foster competition between the students resulting in increased motivation to gain more points and in turn doing better on tests [8].

3.3 Badges

Today many games, social networks and other online applications are using badges as a way to identify users who have mastered a topic or are more driven to finish a task. Where points show incremental progress, one can recognize more isolated achievements by accumulating and displaying badges.

In the AutoEd 2.0 system students are awarded badges for various accomplishments. For example, a student is awarded a badge for being the first student to submit a test. The goal is to have students continuously work towards a badge until they have acquired it resulting in increased motivation and engagement in the system. Along with each badge, a ratio is displayed. This ratio describes how many other users have acquired the badge. By doing this we can challenge students to work towards badges they have not yet completed, resulting in increased motivation to also acquire the said badge. Figure 3.3 below shows the current list of badges available to the user.

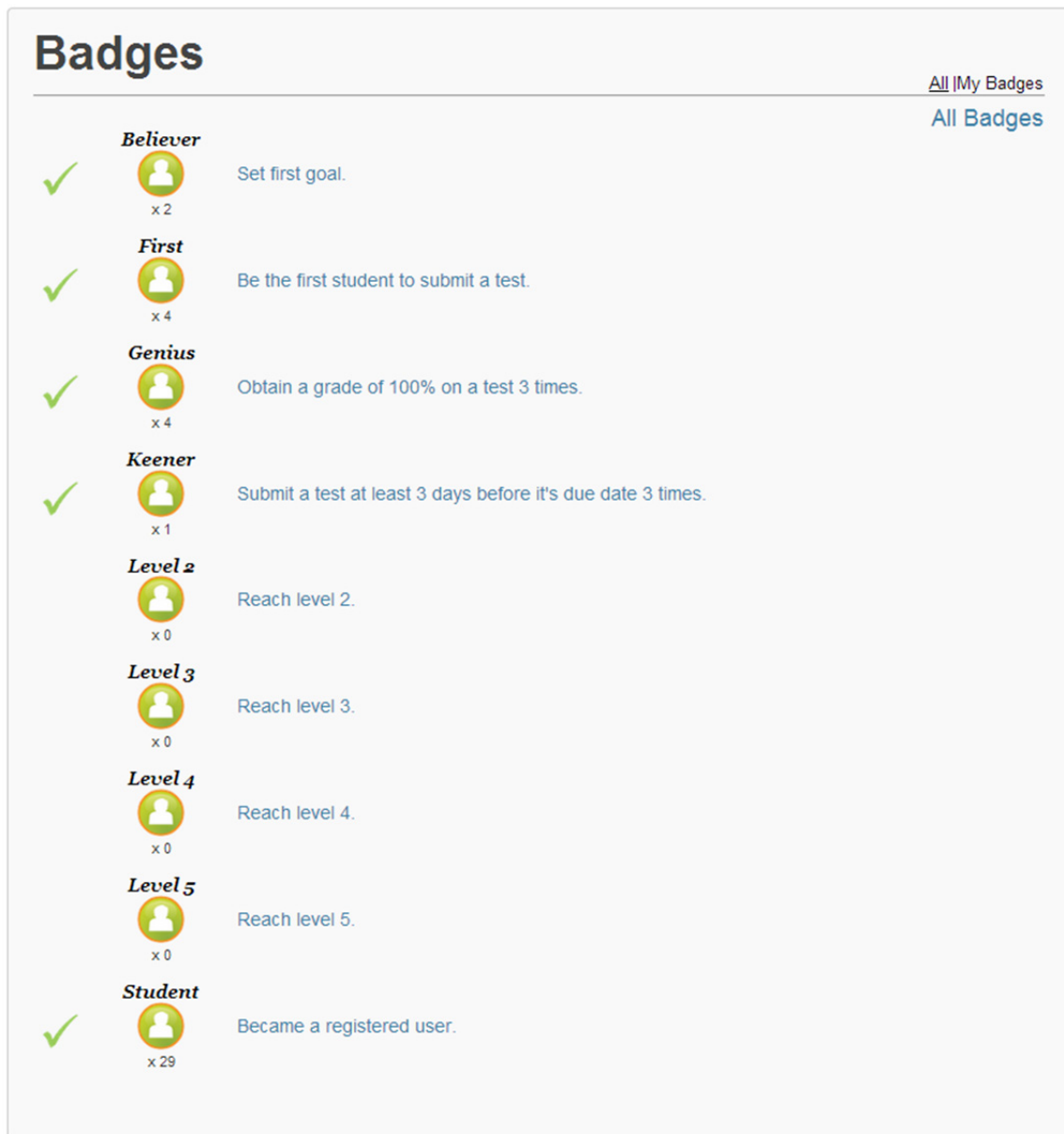


Figure 3.3: A screenshot of all badges available to the user. Notice the badge ratio under each badge. A checkmark denotes that the badge has been completed.

3.4 Student Goals

In most games today there are clearly defined objectives that you must complete in order to win the game or beat another player. The process of working toward a set target and seeing your progression towards this target

can be very exciting. When you finally complete an objective or check of a task you also get a boost of self-confidence. To take advantage of this method a student goals module was implemented. Students stay motivated by setting goals and completing goals at their own pace. In the AutoEd 2.0 system students set their own goals and mark them as completed when they feel the goal has been reached.

3.5 Email Reminders

To help keep students motivated and reduce incomplete assignments an email reminder feature was implemented. In the AutoEd 2.0 system, students are sent emails to remind them of active assignments as well as when they are due along with a reminder to keep working to achieve badges and earn points. An email is sent to a student if there are active tests that the student has not yet completed and if the student has been offline for 3 or more days. If the student wishes to not receive any more emails they can deactivate the email reminders in their profile settings page.

4 High Level Architecture

This section provides a high-level overview of the AutoEd 2.0 system and illustrates how it relates to the previous system.

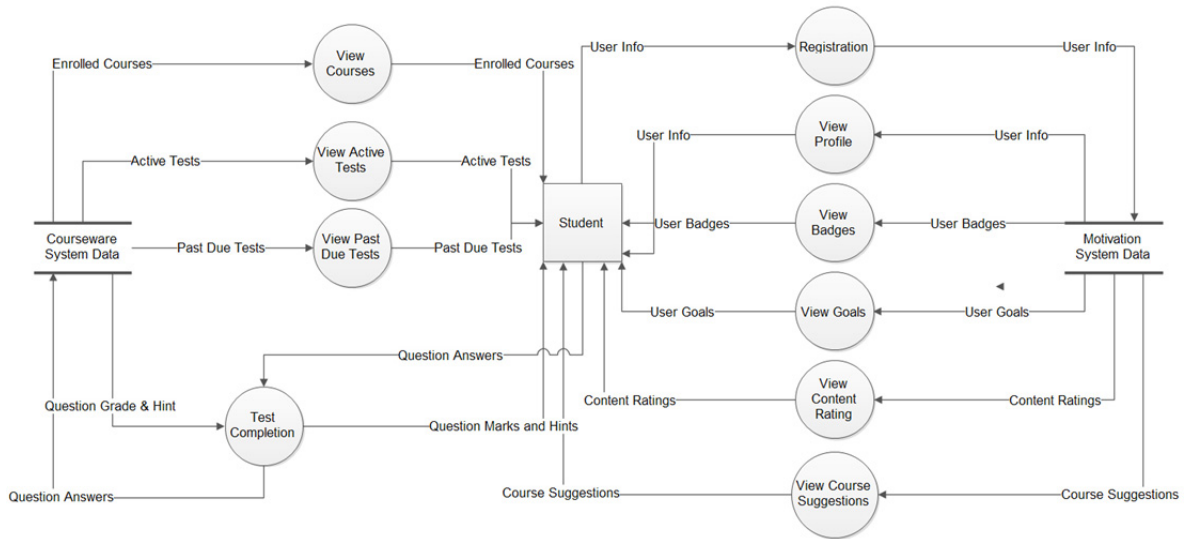


Figure 4: System architecture diagram.

4.1 Modules

4.1.1 Test Completion

AutoEd 2.0 is a web-based courseware system with an integrated user profile system. The courseware system allows students to complete tests for their registered courses online. Professors of these online courses create question templates and using these templates a test is created by generating random questions for the test.

When completing a test, students can save their answers if they decide they need a break or to finish the test at a later date. Like the old system students can also choose to retry questions. However, where the old system would re-generate the entire test, even the questions the student had already answered correctly, and the new system only regenerates the current question template. By doing this, the student does not lose the questions they have answered correctly and can focus on the questions they have answered incorrectly. When submitting an answer to a question students are given immediate feedback whether they answered the question correctly or not.

The system also displays hints to the user if their answer is incorrect to help in the student's next attempt.

3.

A coin lies at the bottom of partly filled rectangular tub of water as shown below. It is observed that a light ray from the coin that travels at the angle of 30 degrees, measured from the water surface, just misses the tub edge at point A. Take the distances $AB = BC = 1.0$ m, and the index of refraction for water, $n = 4/3$.

✓ A) Calculate the critical angle, θ_c , for the water/air interface (You don't need units).

✓ B) Determine the angle θ shown in the diagram.

✗ *Hint: Don't forget units.*

C) What is the depth, h , of the water?

Figure 4.1.1: A screenshot of a sample question. Notice the retry and save button, also the hint “Don’t forget units”.

4.1.2 Profile System

The new and improved AutoEd 2.0 user interface includes a student profile system which allows students to have their own profile page. Each student profile page includes many features to help keep the student motivated. First, the student can view which assignments are due for their currently registered courses as well as how many days are remaining until the assignment's due date. Secondly, students have the ability to set goals to help keep the student motivated and focused in the course. Students are also given the ability to rate courses that they have already completed as well as the courses they are currently enrolled in. Using these ratings a list of course

suggestions are displayed on each student's profile page. Another feature of the profile system is the user transcript section. Here the student can view a transcript of their online courses. Every online course the student has completed is listed in their personal transcript reminding the student what courses they have completed, how many credits they have earned and the rating for that course. The profile system also allows students to search and view the profile pages of other students. By default no information is displayed to other students however each student has the option to select what information they want or do not want made public to other students. Finally, students can configure their email and other personal information in the settings page. Below is a screenshot of a sample student profile page.

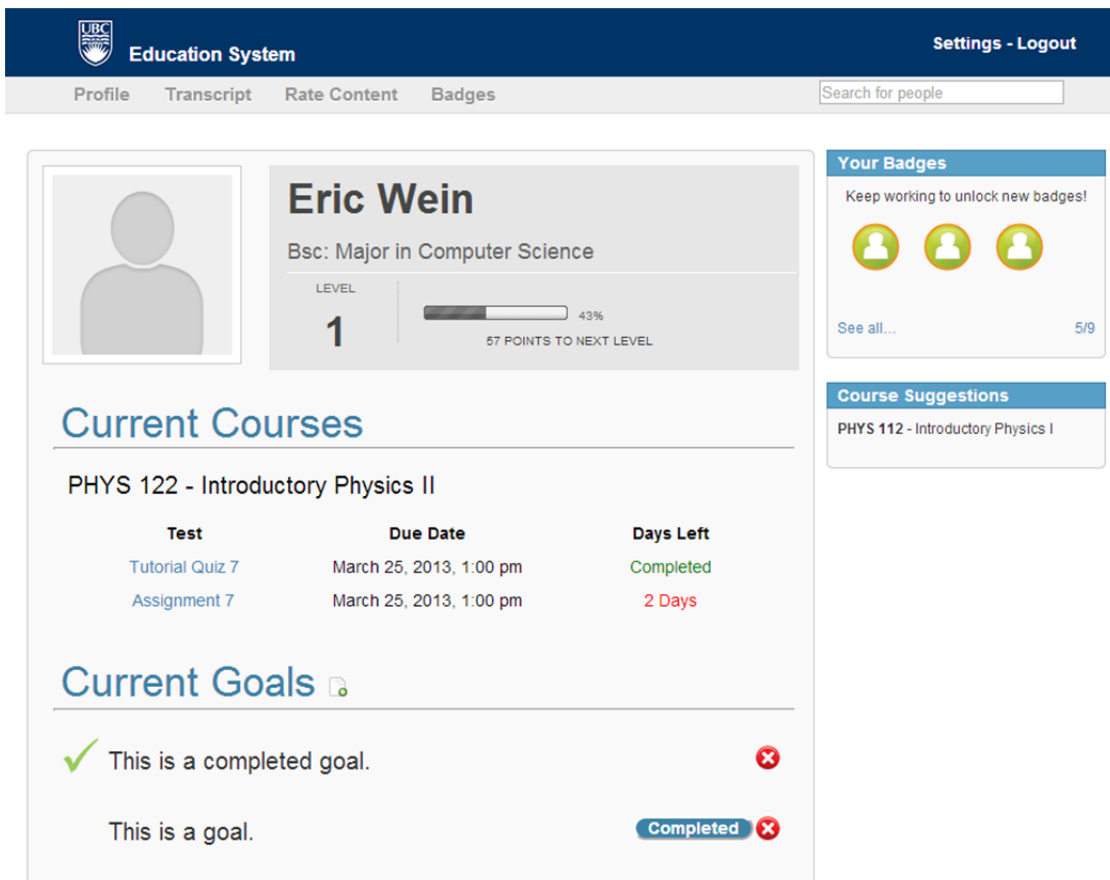


Figure 4.1.2: A screenshot displaying a sample student profile page.

4.2 System Overview

The old AutoEd [2] system uses a Microsoft SQL Server database hosted on a campus server, `cssql.ok.ubc.ca`. This database is used to store information about users and courses as well as test questions, the users' answers and users' test scores. This system was written in the JSP and Java programming languages and is hosted on the campus server `cs-suse-4.ok.ubc.ca`. This code includes the functionality for generating and marking tests.

The AutoEd 2.0 system uses a MySQL Server database hosted locally on the `autoedu.ok.ubc.ca` server. This database is used to store information about the user as well as all the information for the features of the new user interface. This new user interface requires a lot of client-side code, hosted on the `autoedu.ok.ubc.ca` server. Therefore the server side of the AutoEd 2.0 system was implemented using the PHP programming language. PHP is an easy language to use and much easier to integrate with client-side languages such as HTML and JavaScript where the JSP language is not. However, because the old system was written in JSP we needed to develop a way to interact with the old system in order to re-use the generating and marking test features. Our solution to this problem was to develop a PHP wrapper and API.

The PHP wrapper and API are structured as a set of functions which are called throughout the HTML code. This API served as an abstraction layer for the communication and passing of data between the two systems. Using this code structure allowed for separation of responsibility between the server and client-side code. This made the project code very modular and easy to expand and re-use in other parts of the system. The API works by making data requests to the JSP code of the old system, passing the URL of the needed file along with the required parameters. Using these parameters

the JSP generates the content of the requested page which is passed back to PHP as a string. Once the PHP API function has received the file contents string from the old JSP code, the data is parsed to remove the required data.

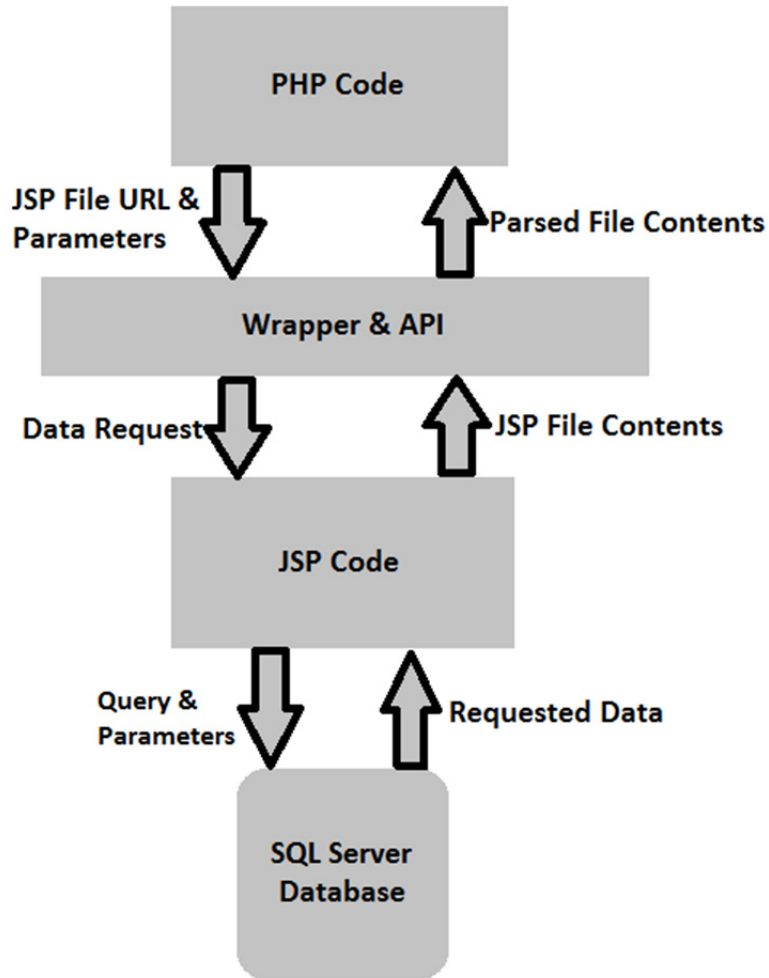


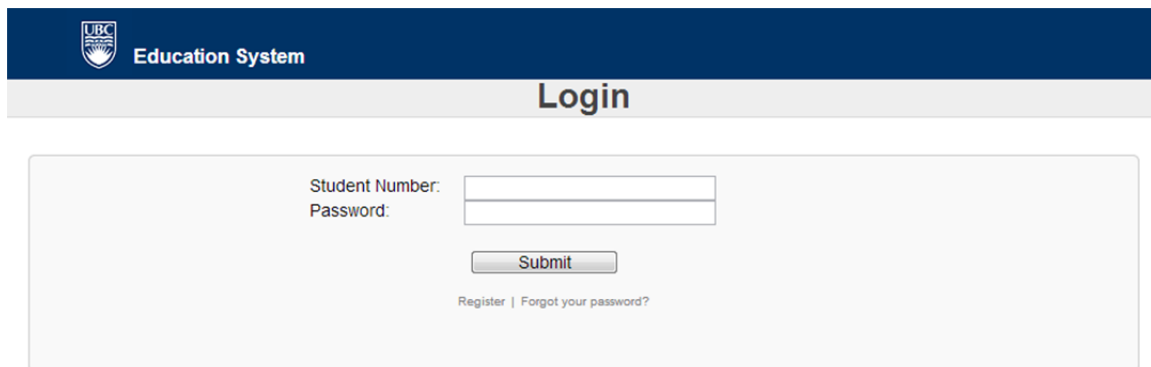
Figure 4.2: A diagram showing how data is passed between the old and new system.

5 User Manual

5.1 Profile System

5.1.1 Logging In

When a student first navigates to autoedu.ok.ubc.ca they are presented with the login screen. Here the user must enter their student number and password that they registered with. The student also has the option to register if they have not already, retrieve their forgotten password and give user feedback.



The screenshot shows the login interface for the UBC Education System. At the top, there is a dark blue header with the UBC logo and the text "Education System". Below the header, the word "Login" is centered in a light gray bar. The main content area is a light gray box containing two input fields: "Student Number:" and "Password:". Below these fields is a "Submit" button. At the bottom of the box, there are two links: "Register" and "Forgot your password?".

Figure 5.1.1: A screenshot showing the system login page.

5.1.2 Registration

Before the student can use the system fully, they must first register in the system. Here the student is required to provide a student number, password, first and last name, a valid email address, the student's birthdate, year of study and finally their degree of study. The student also has the option to upload a profile picture.



Register

Personal Information

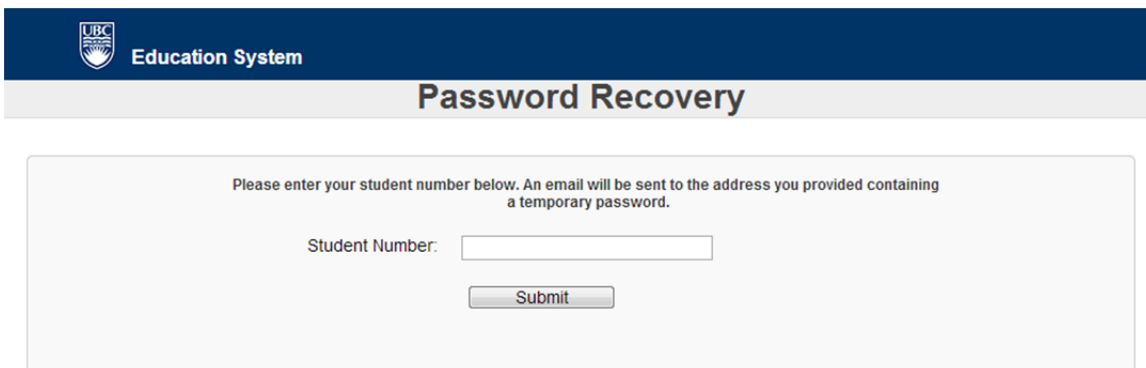
Student Number:	<input type="text"/>
First Name:	<input type="text"/>
Last Name:	<input type="text"/>
Password:	<input type="password"/>
Confirm Password:	<input type="password"/>
Email:	<input type="text"/>
Birthday:	<input type="text" value="mm/dd/yyyy"/>
Year Level:	<input type="text"/>
Degree/Major:	<input type="text"/>


 No file chosen

Figure 5.1.2: A screenshot showing the system registration page.

5.1.3 Password Recovery

If a student has forgotten their password they can go to the password recovery page. Here the student is required to enter their student number. By doing this, an email is sent to the email address associated with the provided student number which the student was required to provide during registration. The email sent to the student contains a randomly generated string which the student can use as a temporary password. Once the user has gained access to their profile page, they can re-assign a new password for their account.



UBC Education System

Password Recovery

Please enter your student number below. An email will be sent to the address you provided containing a temporary password.

Student Number:

Submit

Figure 5.1.3: A screenshot showing the student password recovery page.

5.1.4 Student Profile Page

When the student logs into their account they are presented with their very own profile page. The student profile page is separated into five sections. The first section contains the information about the student including the student's name, degree of study, current level and their progress to their next level. The student can easily add a new profile picture by clicking on the profile picture frame and uploading a new image. The second section displays the student's currently enrolled courses along with the currently active assignments for each enrolled course. For each active test, the test's name, due date and the number of days left until the test's due date are displayed.

The student can start a test by simply clicking on the test name, or by clicking on the course name, the student can view all assignment and tests for the selected course. The third section of the profile page displays the student's goals that they have set. Along with each goal is the option to remove it or mark the goal as completed. A completed goal is denoted with a green checkmark. Students can easily add a new goal by clicking on the add symbol to the right of the "Current Goals" title. The fourth section of the student's profile page displays the badges that the student has unlocked. Here three randomly selected badges that the student has completed are displayed. When a student hovers over each badge, the badge name is displayed. Along with the three badges, the student's badge ratio is displayed. This ratio represents the number of badges the user has acquired to the total number of available badges to be unlocked. The fifth and final section of the profile page is the "Course Suggestions" section. Here the student is displayed the top 3 rated courses in the system that the student has not already completed. Please refer to figure 4.1.2 to view a sample student profile page.

5.1.5 Transcript

Along with the student profile page is the student transcript page. All the courses the student has competed are shown here. For each course, the page shows the course name, the number of credits the student earned, the course status and the rating for the course.

The screenshot shows the UBC Education System interface. At the top, there is a dark blue header with the UBC logo and the text 'Education System'. To the right of the header is a link for 'Settings - Logout'. Below the header is a light gray navigation bar with links for 'Profile', 'Transcript', 'Rate Content', and 'Badges'. A search bar labeled 'Search for people' is also present. The main content area is titled 'Student Transcript' and includes the 'Student ID: 34814103'. Below this is a table with the following data:

Course	Credits	Status	Rating
PHYS 112	3	Completed	4/5

Figure 5.1.5: A screenshot showing the system transcript page.

5.1.6 Rate Content

The AutoEd 2.0 system allows students to rate the courses they are currently enrolled in and the courses they have already completed. For each course the student gives two ratings between one and five, five being the highest. One rating is for the course difficulty and one rating is for the interest of the course. Along with the student's course rating, the overall average rating for the course is displayed.

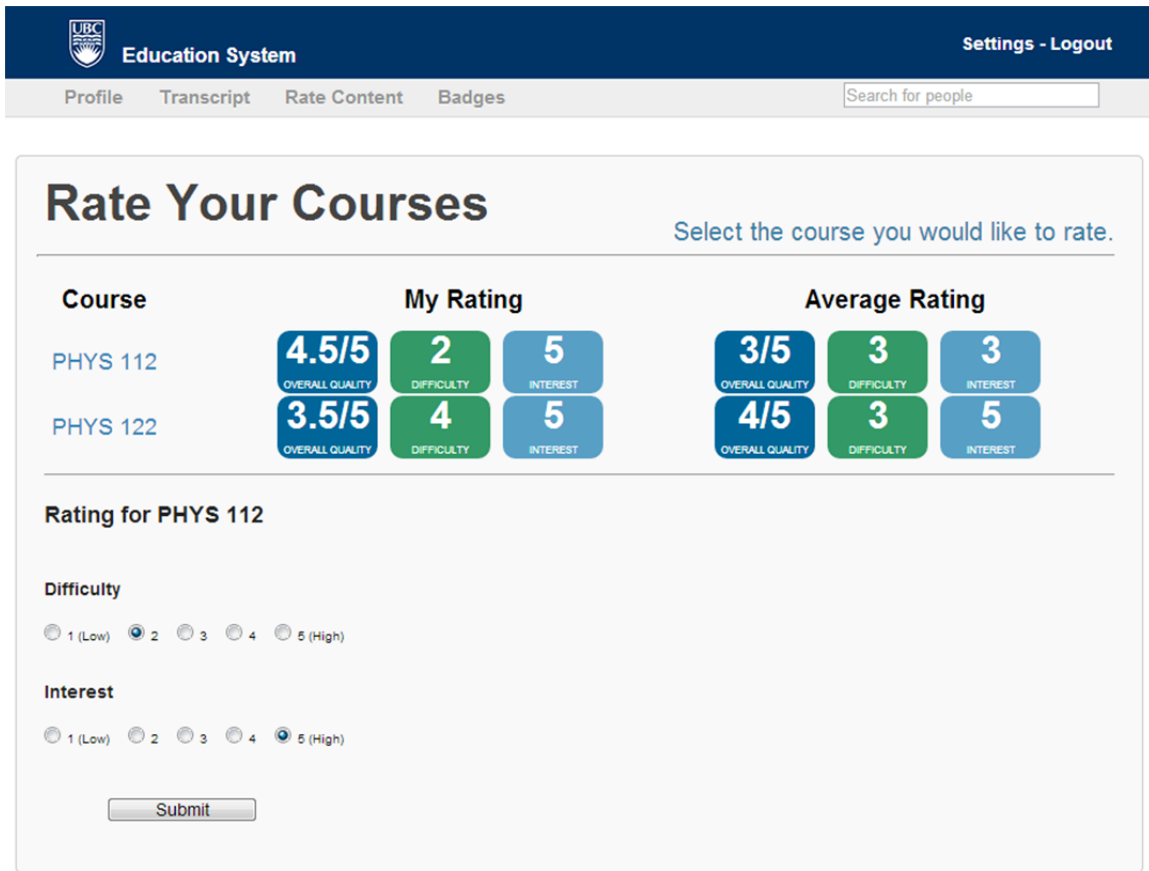


Figure 5.1.6: A screenshot showing the content rating page.

5.1.7 Badges

When students click on the badges tab they are taken to the badges page where the student has three choices. The student can either view only the badges they have achieved, view all the badges available or view both their badges and all available badges in one list. With each badge comes the badge name, a description of the badge, a check mark if the badge has been completed and a number denoting how many students all together have completed the badge. Below is a screenshot showing the badges the student has completed. For a picture showing all available badges please refer to figure 3.3.

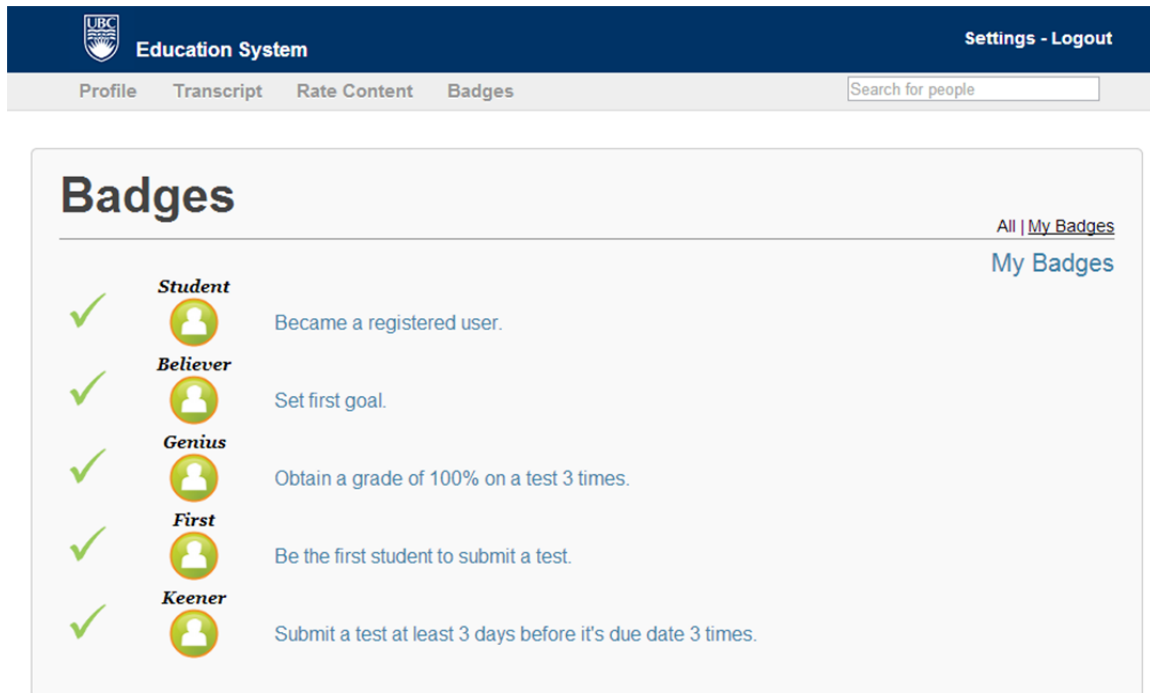


Figure 5.1.7: A screenshot showing the student badges page.

5.1.8 Settings

Students can access the settings page by clicking the settings tab of the system header. Here the student can change their personal information including the name and degree that is displayed on their profile page as well as their email address. Students are also able to change their current password by providing their old password, a new password and a password confirmation. This page also allows the student to choose what information is made public to other students when their profile page is viewed. The student can choose to either hide or show their profile picture, degree, current level, points, enrolled courses, badges and goals by simply checking or unchecking the item. On this page the student also has the option to stop all email messages from the system by clicking the “Remove Email” button as well as the option to deactivate their account by clicking the “Deactivate Account” button.

The screenshot displays the 'Settings - Logout' page of the UBC Education System. At the top, there is a dark blue header with the UBC logo and 'Education System' text on the left, and 'Settings - Logout' on the right. Below the header is a navigation bar with tabs for 'Profile', 'Transcript', 'Rate Content', and 'Badges', and a search bar labeled 'Search for people'. The main content area is divided into three sections:

- Change Information:** Contains input fields for 'First Name', 'Last Name', 'Email', and 'Degree/Major', with a 'Submit' button below.
- Change Password:** Contains input fields for 'Old Password', 'New Password', and 'Confirm password', with a 'Submit' button below.
- Privacy Settings:** Includes a 'Change Privacy Settings' button, a 'Remove Email' button, and a 'Deactivate Account' button. Below these is a section titled 'Privacy Settings' with the instruction 'Select the content that you do not want other users to view.' and a list of checkboxes:
 - Picture
 - Degree
 - Level
 - Courses
 - Goals
 - Badges
 A 'Submit' button is located at the bottom of this section.

Figure 5.1.8: A screenshot showing the system settings page.

5.1.9 System Header

On each page of the user profile system the user is presented with the same page header. Here the user can select from four tabs: “Profile”, which takes the student to their profile page, “Transcript”, which takes the student to their online course transcript, “Rate Content”, which takes the student to the course ratings page and finally the “Badges” tab which takes the student to the badges page. Along with these four tabs are two links, the “Settings” link which takes the student to the settings page and the “Logout” link which logs the student out of the system. Finally the system header contains a search bar where the student can search for other students and view their profile pages if the selected student allows it.

This screenshot shows the system header, which is a dark blue bar at the top of the page. On the left side, it features the UBC logo and the text 'Education System'. On the right side, it says 'Settings - Logout'. Below this bar is a light gray navigation bar containing four tabs: 'Profile', 'Transcript', 'Rate Content', and 'Badges'. To the right of these tabs is a search bar with the placeholder text 'Search for people'.

Figure 5.1.9: A screenshot showing the system header.

5.2 Courseware System

5.2.1 Test Listings

As part of the profile page, all of the student's currently enrolled classes are listed. When the student clicks on one of these course names, the student is taken to the test listings page where they see all active and past due tests for the selected course. Along with each test, the name, due date, grade and the time remaining until the due date is displayed. Students also have a choice to view or hide their past due tests. For each past due test the system displays whether the student had completed the test or whether the test is past due meaning the student did not finish the test.

PHYS 122 - Introductory Physics II			
Test Name	Due Date	Grade	Time Remaining
Tutorial Quiz 7	March 25, 2013, 1:00 pm	100%	Completed
Assignment 7	March 25, 2013, 1:00 pm	64%	2 Days
Past Due Assignments:			
Test Name	Due Date	Grade	Time Remaining
Tutorial Quiz 6	March 13, 2013, 1:00 pm	10%	Completed
Assignment 6	March 13, 2013, 1:00 pm	0%	Past Due
Assignment 5	March 7, 2013, 1:00 pm	16%	Completed
Tutorial Quiz 5	March 4, 2013, 1:00 pm	44%	Completed
Tutorial Quiz 4	February 11, 2013, 1:00 pm	83%	Completed
Assignment 4	February 11, 2013, 1:00 pm	4%	Completed
Tutorial Quiz 3	February 4, 2013, 1:00 pm	100%	Completed
Assignment 3	February 4, 2013, 1:00 pm	0%	Past Due
Assignment 2	January 26, 2013, 1:00 pm	23%	Completed
Tutorial Quiz 2	January 26, 2013, 1:00 pm	0%	Past Due
Tutorial Quiz 1	January 18, 2013, 11:00 pm	0%	Past Due
Assignment 1	January 18, 2013, 11:00 pm	0%	Past Due

[Hide Past Due Assignments](#)

Figure 5.2.1: A screenshot showing the test listing for PHYS 122.

5.2.2 Active Test

When the student clicks on an active test the student is immediately taken to that test. If the student had already previously attempted the test, the test marks are generated and displayed. For each question the student has the option to submit the question for marking, clear the question, save the question if they wish to work on it later or retry the question. When the student chooses to retry a question a new question template is randomly generated for the current test. However the core structure of the question remains and only the question values change to give the student a fresh start on the question without making the question any easier or harder. When the student submits a question, the question is immediately graded and marks are returned in real time along with a hint if the student answered incorrectly. The student can also choose to submit the entire test at once, clear the entire test or save the entire test. For a screenshot of a sample active test question please refer to Figure 4.1.1.

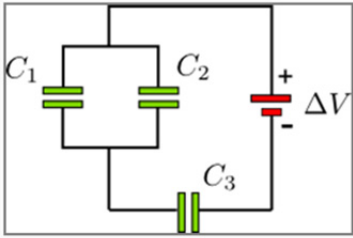
5.2.3 Past Due Test

When the student clicks on a past due test, the student is immediately taken to the selected test. Along with the test all the marks for each question are displayed. When viewing a past due test, students can view the answers they submitted and whether they are correct or not. However all input fields and buttons are disabled therefore the student cannot change or submit new answers.

Tutorial Quiz 3

1.

Three capacitors $C_1 = 3\mu\text{F}$, $C_2 = 3\mu\text{F}$ and $C_3 = 3\mu\text{F}$.



✓ A) What is the equivalent capacitance, C_{eq} of the circuit? To enter a number in scientific notation use the format: 5E-6.

✗ *Hint: Don't forget units.*

B) What voltage ΔV is required to store $7.2 \text{ E-}5 \text{ C}$ of charge on the C_{eq} ?

2.

A parallel plate capacitor has an area of $5.00 \text{ E-}4 \text{ m}^2$ and a plate separation of $d = 1.00 \text{ E-}3 \text{ m}$. The material between the plates is air.

✓ What is its capacitance?

Figure 5.2.3: A screenshot showing a sample past due test.

6 Development

6.1 Challenges

Throughout the development of the AutoEd 2.0 system two main issues came to the surface. The first problem came from showing the student goals. The original idea was to show the student's progression in achieving their goal. However because students create their own goals there is no way to determine what the goal is or what needs to be done to achieve the given goal. Therefore the system now allows the student to mark their own goals as completed when they feel they have achieved the goal. Another major problem came from integrating the old AutoEd system with the new AutoEd

2.0 system. Because the old system was written in the JSP programming language and the new system was written in PHP programming language, the functionality of the old system could not be copied into the new system. In order to resolve this issue a PHP wrapper and API were created to act as an abstraction layer for the passing of data from the old system to the new.

7 Results and Feedback

The AutoEd 2.0 system was available for students of two first year physics courses for the last three weeks of the winter semester of the 2012/2013 school year. Over this period of time the system had 43 registered users along with 27 completed badges and 394 awarded points. These numbers show that for the small subset of students who have switched to the new system they have been very active in the AutoEd 2.0 system. For the last two weeks of the semester the system was gathering student feedback such as: “The new system is fun with all of the new levels and badges to earn.”(Anonymous) and “the system is very well designed and easy to use!”(Anonymous). From this feedback, students give the impression that they have really enjoyed the system and believe it is a huge improvement over the old AutoEd system. However, because the system has only been available for three weeks, the data set gathered thus far is too small to make any conclusions whether or not the AutoEd 2.0 system has reached its goal of increasing student performance in online courses. Though it seems the students who have been using the AutoEd 2.0 system really enjoy using it and far prefer it over the old system. However the system still needs to go through user studies and a HCI evaluation.

8 Conclusions

AutoEd 2.0 is a valuable interactive online learning tool to help keep students motivated and engaged in their online courses. The system was used by two first year physics classes for 3 assignments. Feedback given by the students was very good. Most students really like the new badge, points and level systems, with most comments resulting in the student choosing the new system over the old.

9 Future Work

Due to time constraints not all the planned features were implemented in the AutoEd 2.0 system. One feature that was desired was a student leader board. The student leader board would be available to all students and teachers of the system. The board would show all the students and their rankings with the students' rank being determined by the number of badges the student had completed and the number of points they had acquired. The leaderboard would only show the student if the student allows it in the settings page. Including such a leaderboard would increase the competitive nature of the badge and points systems resulting in the students working harder to be ranked higher on the leaderboard. Another feature that was not implemented was a student message board. Here students and teachers would have the ability to post comments about the system, current tests or have general discussions. By including this feature the system would allow for more social interaction and collaboration between the students adding to the relaxed feeling of the system. The third feature that was not included due to time constraints was a new user interface for the AutoEd teacher system. Like the old student interface the layout and overall look of the system is very dull, cluttered and confusing to use. Therefore a new, clean and intuitive teacher interface is needed. The final feature that was not included

was a student store where students would use the points they have earned to buy various perks. For example, a student could use their points to extend the due date of a test or to buy a “redo attempt” of a past due test they did not complete. In order for this feature to be included some collaboration and acceptance would be needed by the professors of the online courses.

Appendix

Table of Contents

Database DDL Script	28
PHP Files.....	29
config.php	29
connectDB.php	29
header.php	29
ajax.php	29
register.php	30
login.php	30
logout.php	30
index.php	30
search.php	30
profile.php.....	30
badges.php	31
rate.php.....	31
settings.php	31
scheduledTask.php.....	31
feedback.php.....	31
sendPassword.php.....	31
tests.php	32
userTest.php.....	32

templates/header.php.....	32
templates/footer.php	32
includes/crypto.php	32
includes/email-validator.php	33
templates/phpaes.php	33
templates/functions.php.....	33
api/api.php	33
api/tests.php	33
api/users.php	34
JavaScript Files.....	38
js/bramus/jsProgressBarHandler.js.....	38
js/bramus/prototype.js.....	38
scripts/jquery.js	38
scripts/question.js.....	38
scripts/script.js	39

Database DDL Script

```
1 CREATE TABLE IF NOT EXISTS `badges` (  
2   `bid` int(11) NOT NULL AUTO_INCREMENT,  
3   `bname` varchar(150) NOT NULL,  
4   `description` varchar(255) NOT NULL,  
5   `picture` varchar(150) NOT NULL,  
6   PRIMARY KEY (`bid`)  
7 ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=12 ;  
8  
9 CREATE TABLE IF NOT EXISTS `courses` (  
10  `cid` int(11) NOT NULL,  
11  `cnum` varchar(8) NOT NULL,  
12  `department` varchar(150) NOT NULL,  
13  `cname` varchar(255) NOT NULL,  
14  `credits` int(11) NOT NULL,  
15  PRIMARY KEY (`cid`)  
16 ) ENGINE=InnoDB DEFAULT CHARSET=utf8;  
17  
18 CREATE TABLE IF NOT EXISTS `enrolled` (  
19  `uid` char(8) NOT NULL,  
20  `cid` int(11) NOT NULL,  
21  `grade` varchar(50) NOT NULL,  
22  `progress` int(11) NOT NULL,  
23  `rating` double NOT NULL,  
24  `difficulty` double NOT NULL,  
25  `interest` double NOT NULL,  
26  `completed` char(1) NOT NULL,  
27  PRIMARY KEY (`uid`,`cid`),  
28  KEY `cid` (`cid`)  
29 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
30  
31 CREATE TABLE IF NOT EXISTS `feedback` (  
32  `id` int(11) NOT NULL AUTO_INCREMENT,  
33  `system` varchar(10) NOT NULL,  
34  `feedback` text NOT NULL,  
35  PRIMARY KEY (`id`)  
36 ) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=9 ;  
37  
38 CREATE TABLE IF NOT EXISTS `goals` (  
39  `gid` int(11) NOT NULL AUTO_INCREMENT,  
40  `uid` char(8) NOT NULL,  
41  `description` varchar(255) NOT NULL,  
42  `progress` int(11) NOT NULL,  
43  PRIMARY KEY (`gid`,`uid`),  
44  KEY `uid` (`uid`)  
45 ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=22 ;  
46  
47 CREATE TABLE IF NOT EXISTS `points` (  
48  `id` int(11) NOT NULL AUTO_INCREMENT,  
49  `uid` char(8) NOT NULL,  
50  `score` int(11) NOT NULL,  
51  `tid` int(11) DEFAULT NULL,  
52  `description` varchar(255) DEFAULT NULL,  
53  PRIMARY KEY (`id`)  
54 ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=107 ;  
55  
56 CREATE TABLE IF NOT EXISTS `privacy` (  
57  `uid` char(8) NOT NULL,  
58  `picture` int(11) NOT NULL,  
59  `degree` int(11) NOT NULL,  
60  `level` int(11) NOT NULL,  
61  `courses` int(11) NOT NULL,  
62  `goals` int(11) NOT NULL,  
63  `badges` int(11) NOT NULL,  
64  PRIMARY KEY (`uid`)  
65 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
66  
67 CREATE TABLE IF NOT EXISTS `questions` (  
68  `id` int(11) NOT NULL AUTO_INCREMENT,  
69  `hash` varchar(255) NOT NULL,  
70  `value` text,  
71  PRIMARY KEY (`id`)  
72 ) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;  
73  
74 CREATE TABLE IF NOT EXISTS `tests` (  
75  `uid` char(8) NOT NULL,  
76  `tid` int(11) NOT NULL,  
77  `ctid` int(11) DEFAULT NULL,  
78  `completed` varchar(10) NOT NULL,  
79  UNIQUE KEY `uid` (`uid`,`tid`,`ctid`)  
80 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
81  
82 CREATE TABLE IF NOT EXISTS `user` (  
83  `uid` char(8) NOT NULL,  
84  `remote_uid` int(11) DEFAULT NULL,  
85  `firstname` varchar(100) NOT NULL,  
86  `lastname` varchar(100) NOT NULL,  
87  `password` varchar(100) NOT NULL,  
88  `email` varchar(100) NOT NULL,  
89  `birthdate` varchar(15) NOT NULL,  
90  `year` int(11) DEFAULT NULL,  
91  `degree` varchar(100) NOT NULL,  
92  `level` int(11) NOT NULL,  
93  `picture` varchar(150) NOT NULL,  
94  `dateCreated` datetime NOT NULL,  
95  `logoutTime` datetime DEFAULT NULL,  
96  PRIMARY KEY (`uid`)  
97 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
98  
99 CREATE TABLE IF NOT EXISTS `userbadges` (  
100  `uid` char(8) NOT NULL,  
101  `bid` int(11) NOT NULL,  
102  `count` int(11) NOT NULL DEFAULT '-1',  
103  PRIMARY KEY (`uid`,`bid`),  
104  KEY `bid` (`bid`)  
105 ) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

Figure AI: A screenshot of the database DDL script.

PHP Files

config.php

Summary: Instantiates a global configuration array to store important information for use throughout the system.

connectDB.php

Summary: Handles setting up connections to the MySQL (\$conn) and SQLServer (\$remote_conn) databases. This page also sets up the PDO abstraction layer to handle all calls to the database to prevent injection attacks by using prepared statements.

header.php

Summary: Handles user session variables and decides whether the user is logged in or not.

ajax.php

Summary: Handles all Ajax calls from the JavaScript files. This file is structured as one large switch statement switching on the “method” parameter that is used in all Ajax calls. The method parameter is used to select the correct actions to perform.

- **case “regen_question”** – used to generate a new version of a given question and returns the appropriate markup.
- **case “save_question”** - saves a question and returns if the question was correct (and the hint).
- **case “submit_test”** – used to submit an entire test for grading as well as give points and badges relating to tests.
- **case “removeEmail”** – used to remove the users email from the system.

- **case “deactive”** – deactivate the user’s account from the system meaning to remove all content about the user from the system.
- **case “completegoal”** – used to mark a goal as completed.
- **case “deletegoal”** – used to delete users goal from database.

register.php

Summary: Displays the registration form and handles user registration which includes profile picture uploads.

login.php

Summary: Displays the login form and handles user login.

logout.php

Summary: Handles user logout by destroying all session variables.

index.php

Summary: Displays the user profile page, including the user’s information, points and level, profile picture, current courses, active tests, user goals, user badges and suggested courses.

search.php

Summary: Handles all search bar activity, as a student types into the “Search People” search bar content is immediately displayed.

profile.php

Summary: This page is called when the user searches and views the profile page of another student. This page displays the allowed information for the selected user. Each section of the profile page is denoted by a flag, either 0 or 1. If a section is labeled 0 then the student allows this content to be publicly displayed, otherwise it is not displayed.

badges.php

Summary: Selects all the badges from the system as well as the badges the user has completed and displays these badges to the user. This page also handles the badge toggle events.

rate.php

Summary: Selects all the courses the user has completed or is currently taking and displays the users rating for each course as well as the average rating for this course. When the course name is clicked, a panel is displayed which allows the student to submit a new rating.

settings.php

Summary: Displays the settings form where users can change their user information including their name, degree and email address. The user can also change their password, change their public information, remove their email or deactivate their account.

scheduledTask.php

Summary: This file sends emails to students if they have been inactive on the system for 3 or more days while there are active tests.

feedback.php

Summary: Contains two text areas one for the old system and one for the new system to allow for user feedback.

sendPassword.php

Summary: Generates a random temporary password for the user and sends it to the email address the student provided. This temporary password is stored as the user's password until the student changes the password on the settings page.

tests.php

Summary: Generates the users active tests and past due tests and displays them in a table. For each test, the test grade is computed as well as how many days are left to the due date or whether it is past due. If a test is active and the student has achieved a grade of 100% then the test is marked as completed.

userTest.php

Summary: This page handles the loading and activity of completing a test. By using the \$tid of the tests a set of random questions are generated for the user to complete. If the student has already attempted the test then the question marks are generated and displayed. This page contains two important variables, \$getmarks and \$active. If \$getmarks is true then the page is required to display the users marks. If \$active is true then the page must allow the user to submit new answers. If \$active is false then all input fields and buttons are disabled.

templates/header.php

Summary: This page contains the header information for the HTML pages to avoid using the same code throughout the site and to keep the system modular. This file includes all the style sheets and JavaScript files needed for the site as well as the system toolbar.

templates/footer.php

Summary: This page contains the footer information for the HTML pages to avoid using the same code throughout the site and to keep the system modular.

includes/crypto.php

Summary: Handles all password hashing for the system. All hashing is done using a salted SHA256 hash.

includes/email-validator.php

Summary: This file is used to validate user email addresses. Each email address must be less than 256 characters with a minimum of 3 characters.

templates/phpaes.php

Summary: AES Cipher function for encrypt function input.

templates/functions.php

Summary: contains the function generatePassword(\$length=10) to generate a random password for the user.

api/api.php

Summary: This file includes the tests.php and user.php files so the system remains modular. By doing this the other files of the system only needs to require this file.

api/tests.php

Summary: This file contains all the functions needed for generating, marking and submitting user tests.

- **generateTID(\$ttid, \$remote_ID)** – Using the test template id (\$ttid) and the students remote id (\$remote_ID) a new test id (\$tid) is generated for the new test attempt. Returns a new test id (\$tid).
- **prepareJSON(\$input)** – This function is used to fix strange behaviour with JSON from the server side. Returns a JSON string.
- **checkAnswer(\$tid, \$index, \$answer, \$save = “true”)** - This functions takes a test id, question index, answer and a save value and

checks if the given answer is correct. Returns an array containing the users mark, and hint.

- **getQuestion(\$tid, \$index, \$new=false)** – Using a test id, and question index this function returns a single question from a test template ordered by index. Returns pre-generated mark-up for a question generated from the JSP code of the old system.
- **isSerialized(\$string)** – This functions checks if a string is a serialization or not.
- **getQuestionsByTest(\$tid, \$num=null)** – This function returns the list of questions for a given test id. The questions are represented as a dictionary with a property called 'html'. This HTML is as simply as possible, and is generated on the JSP side of the old system.
- **getDatabaseDate(\$timestamp = null)** – This function returns the date in the appropriate database format, just as an abstraction in case the database is changed in the future.
- **getTestMark(\$tid, \$remote_id)** – This function returns the grade for the test with the given test id (\$tid).
- **getTestsByCouse(\$courseID, \$userID, \$condition)** – This function generates all active and past due tests along with all their attempts for the given course id and user id.
- **getTTID(\$tid)** – This function returns the test template id for the given test id.

api/users.php

Summary: This file contains all the functions needed for the features of the new user interface and motivation system.

- **checkLogin(\$uid, \$password_attempt)** – This function checks if the student has provided the correct password for the given user id. If

the password match then a new session is started and the student is redirected to the index.php page.

- **register(\$uid, \$firstname, \$lastname, \$password, \$birthdate, \$email, \$year, \$degree, \$file, \$quiet=false)** – This function takes the information the user submitted in the registration form and inserts content into the database.
- **uploadPicture(\$uid, \$file, \$update)** – This function is used to handle profile picture uploads. The given file to be uploaded (\$file) must be of type jpeg or png and can be at most 5mb in size. When a student first uploads a profile picture a personal directory is created to store all profile picture for the user. These directories are stored in the uploads directory.
- **sendPassword(\$uid)** – This function generates a new random password and sends this new password via email to the user if id equal to \$uid.
- **userInfo(\$uid)** – Returns an array containing all the users information from the database.
- **checkBadge(\$uid, \$bid)** – This function checks if the user with the given id has completed the badge with id equal to \$bid. If the user has not completed the badge then the database is updated to show the user has completed the badge.
- **addCourse(\$uid, \$course)** – This function adds a new enrolled course for the user.
- **getEnrolled(\$uid)** – This function returns all the courses the user is currently enrolled in.
- **getGoals(\$uid)** – This function returns all the user goals and whether they are completed or not.
- **getBadges(\$uid, \$id)** – This function returns the user's and system badges. If \$id is equal to "3" then 3 random badges are returned to the

user as seen on the users profile page. Otherwise all the student's badges and all the possible badges are returned.

- **recomendedCourses(\$uid)** – This function returns the three highest rated courses that the student has not yet completed or is currently enrolled in.
- **deleteGoal(\$id)** – This function removes the goal with the given id.
- **completeGoal(\$uid, \$gid)** - This function marks the goal with the given id as completed for the given user id.
- **logout(\$uid)** – This function handles system logout by destroying all session variables for the current user.
- **privacySettings(\$uid)** – This function returns the privacy settings for the user. An item either has the value 0 or 1. If an item has a value of 0 then the student allows for this content to be made public, otherwise the content is not made public.
- **rateCourse(\$cnum, \$dept, \$rating, \$interest, \$diff, \$uid)** – This function updates the students rating for the given course (\$cnum, \$dept) with the given ratings for the given student id.
- **getCourseInfo(\$cid)** – This function returns all the content for the given course.
- **getCourses(\$uid)** – This function returns all the courses the student is currently enrolled in as well as all the courses the student has already completed.
- **getRating(\$cid)** – This function returns the average rating for the given course.
- **scheduledTask()** – This function is ran every day at noon and checks for inactive students. Students are inactive if they have been away from the site for 3 or more days while there are active tests. This function sends an email to each inactive student.
- **search(\$firstname, \$lastname)** – This function returns the markup for the “Search People” search bar results.

- **transcript(\$uid)** – This function returns an array containing the information for the student online transcript.
- **getLevel(\$uid)** – This function aggregates all the given users points and determines what level they are at and returns this level.
- **getPoints(\$uid)** – This function returns the total number of points for the given user.
- **getProgress(\$uid)** – This function returns the users progress to their next level.
- **addPoints(\$uid, \$points, \$tid, \$description)** – This function gives points to users as well as a description of what the user did to earn the points.
- **deactivateAccount(\$uid)** – This function handles the removing of all the users content from the database when they deactivate their account.
- **badgeRatio(\$uid)** – This function returns the users badge ratio, this ratio is the number of badges the user has completed over the total number of badges.
- **checkFirst(\$uid, \$ttid)** – This function checks if the given student was the first student to submit the given test.
- **checkSubmissionGoal(\$uid, \$ttid)** – This function returns the number of days before the due date that the given test was submitted.
- **removeEmail(\$uid)** – This function removes the users email address from the system. The system requires an email address for the user so a system email address was created (ubccoursewaretrash@gmail.com) to take the place of the users email address. When an email is sent to this email address, the email is immediately deleted to avoid any space issues.
- **userFeedback(\$id, \$feedback)** – This function enters the users feedback into the database for the given system id.

JavaScript Files

js/bramus/jsProgressBarHandler.js

Bramus [4] is an open source JavaScript package for web development. This package includes the jsProgressBarHandler.js file which allows for various, easily adaptable progress bars.

js/bramus/prototype.js

Prototype is an open source JavaScript framework which “adds useful extensions to the browser scripting environment” [5].

scripts/jquery.js

“jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers.” [6].

scripts/question.js

Summary: This file handles all activity for a completing a test. This includes, submitting, saving, retying and clear questions.

- **clearQuestion(id)** – This function clears all input fields for the question with the given id.
- **regenQuestion(index, test_id)** – This function generates a new question for the current index and test by making an Ajax call to ajax.php requesting a new question.
- **markQuestion(i, tid)** – The function marks the *i*-th question for the given test.
- **saveQuestion(i, tid)** – This function saves the *i*-th question for the given test.

- **mySaveQuestion(i, tid, save)** – For each input field for the *i*-th question for the given test we make an Ajax call to ajax.php passing the users answer to be saved for later.
- **ajaxCall(url, f)** – This function handles all Ajax calls. The data is passed to the given url and the function “f” is executed when the Ajax call returns.
- **submitTest()** – This function submits all input fields for the current test.
- **getMarks(type)** – This function returns the marks for each question of the current test. If type is “not_active” we disable all input fields and buttons.
- **clearAll()** – This function clears all input fields for the current test.
- **saveAll()** – This function saves all input fields for the current test.

scripts/script.js

Summary: This file is used to handle all collapsing panels.

- **toggleDiv(divToShow)** – This function handles all showing and hiding for the given div element.
- **expand(divName)** – This function expands the given div element.
- **collapse(divName)** – This function collapses the given div element.
- **showDiv(divName)** – This function makes the given div element visible.
- **getCourse()** – This function sets the hidden input field values for the selected rating values.
- **showRating()** – This function is used to expand the rating form for the select course and load the correct rating values.
- **toggleBadges()** – This function handles the toggling of the two badge panels in badges.php.

- **setProgress(barName, amount)** – This function is used to set the percentage of a progress bar.
- **getResults(str)** – This function returns the search results for the given string.
- **showDelete(id)** – This function handles the showing and hiding of the delete buttons for goals.
- **ajaxCall(url, f)** – This function handles all Ajax calls. The data is passed to the given url and the function “f” is executed when the Ajax call returns.
- **deactivate()** – This function sends an Ajax request to deactivate the current users account.
- **removeEmail()** – This function sends an Ajax request to remove the current users email account.
- **deleteGoal(id)** – This function sends an Ajax request to remove the current goal.
- **completeGoal(id)** – This function sends an Ajax request to update the database to make a goal as completed and if the Ajax request returns true then display a green checkmark with the goal.

References

- [1] Aplia. <http://www.aplia.com/>
- [2] Pushak, Alyosha. [*"AutoEd: An Online Assignment Generation and Marking System"*](#), Undergraduate Honours Thesis UBC Okanagan 2011.
- [3] Gradiance. <http://www.gradiance.com/>
- [4] Bramus (2011). <http://www.bram.us/>
- [5] Prototype (2012). <http://prototypejs.org/>
- [6] JQuery (2013). <http://jquery.com/>
- [7] S. Deterding, S. Björk, L. E. Nacke, D. Dixon, E. Lawley. [*"Designing Gamification: Creating Gameful and Playful Experiences"*](#), (2013).
- [8] L. S. Ferro, S. P. Walz, [*"Like this: How game elements in social media and collaboration are changing the flow of information."*](#), (2013).