Technological Educational Institute of Crete

Department of Informatics Engineering

Bachelor Thesis

Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή Εργασία

Title: *On-line quiz generator for educational purposes in JavaEE*

Papadopoulou Anna
A.M 2862

Supervisor: Dr. Nikolaos Vidakis

ACKNOWLEDGEMENTS

Firstly, I would like to express my gratitude to my thesis mentor, Professor Nikos Vidakis for his help, patience and guidance over my years of study at T.E.I of Crete.

I would also like to thank my advisor and friend Gabor Kaposi for sharing his knowledge on such amazing technologies and supporting me with enthusiasm!

Abstract

This project introduces a web-application called MyQuizDB that allows the user to generate and play quizzes as well as sharing them with other users and validate their results. Its main purpose is to help teachers test their students and students test themselves, in a fast, futuristic and enjoyable way.

After account creation, any user can

1) Create a quiz
2) Share it with other users
3) Validate their results
4) Create pools of questions for later use
5) Add questions to a pool while playing

The application is created in Eclipse[1] development environment. Its main development language is JavaEE. It also uses the open source UI framework Primefaces and the front-end framework Bootstrap. All diagrams were created with Visual Paradigm[2] and illustration were developed with Adobe Photoshop[3].

Σύνοψη

Αυτή η εργασία είναι μία διαδικτυακή εφαρμογή που ονομάζεται MyQuizDB. To MyQuizDB επιτρέπει στον χρήστη να δημιουργεί και να παίζει κουίζ καθώς και να τα μοιράζεται με άλλους χρήστες και αξιολογεί τα αποτελέσματά τους. Κύριος στόχος της παρούσας εργασίας είναι να βοηθήσει τους δασκάλους στον έλεγχο γνώσεων των μαθητών τους αλλά και τους μαθητές στον αυτοέλεγχό τους. Η ανάπτυξη αυτής της εφαρμογής προσπαθεί να μετατρέψει την διαδικασία διαγωνισμάτων σε μία γρήγορη, σύγχρονη και ευχάριστη διαδικασία.

Μετά τη δημιουργία λογαριασμού, οποιοσδήποτε χρήστης μπορεί να:

1) Δημιουργήσει κουίζ
2) Μοιραστεί τα κουίζ του/της με άλλους χρήστες
3) Ελέγξει τα αποτελέσματά τους
4) Δημιουργήσει τράπεζες ερωτήσεων (pools) για μετέπειτα χρήση
5) Προσθέσει μία ερώτηση σε κάποιο pool κατά το παίξιμο κουίζ

Η εφαρμογή γράφτηκε στο περιβάλλον ανάπτυξης λογισμικού Eclipse. Η κύρια γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η JavaEE. Επίσης χρησιμοποιήθηκε το UI framework ανοιχτού κώδικα Primefaces καθώς και το front-end framework Bootstrap. Όλα τα διαγράμματα δημιουργήθηκαν στο Visual Paradigm και όλες τις εικόνες με στο Adobe Photoshop.

Table of Contents

Table of Figures

1. Useful Definitions & State of Art

1.1 Why Quiz?

Education is the cornerstone of human society and an integral part of one's life. It's methods, shapes and forms keep changing to match every era's needs, continuing the eternal loop of evolution. In the present day, the traditional method of the educational process follows a teacher-oriented and classroom-based model. Like most of us know, this means that the teacher lectures and guides while the student is expected to listen and learn. Studying techniques mostly involve re-reading, reviewing material again and again until the information is stored in the student's memory. Finally comes the evaluation of knowledge in the form of a test. Although this method is effective for some, it takes an exhaustive effort and a tremendous amount of time to learn. And how much does the student remember during or after taking the exam? Have you ever spent hours re-reading the book but failed on the test? Well everybody has and here is why.

Let's suppose that a person goes for shopping holding a big empty basket. Some groceries can be found easily. Some others are hard to spot so the shopper walks the halls again and again to find them. Other items are too high to reach so he climbs a ladder to get them. No matter how easy or hard it was, the persistent customer manages to get all groceries and places them inside the basket. Later on, he takes each item out and uses it. People tend to think of memory as a basket full of information and that learning works like grocery shopping, except it doesn't. Just like the above example, teachers and learners focus on storing the information into the memory, thinking that once it's in the "basket", it can be used when it's needed. But memory works in a different way. According to Henry L. Roediger, III[1] the emphasis shouldn't be on acquiring the information but on **retrieving** it and the actual part of pulling out the same information from the memory, over and over again is what makes it retrievable on request.

Although the idea of the so called **Testing effect** or **Retrieval Practice[4]**, has been discovered by many people over the centuries, the education system still holds on to conservatism left-overs, meaning that the exam is a grade-giving process and stress point to both students and teachers. In an alternative way, testing as part of studying has proven to be more efficient than re-reading or reviewing the same information. The reason behind this is under debate. One theory says, that every stored memory is attached to a particular state of sensations and retrieving the same material connects it to more sensations. So increasing the amount of sensations that a memory is connected to, helps to make it available when it's needed. In any case, quizzing can not only be used as an educational tool but also as a way to keep one's mind sharp while having a good time.

[1] Principal Investigator at Memory Lab and Psychology Professor at the Department of Psychological & Brain Sciences, Washington University, St. Louis

1.2 Existing Quiz Generating Systems

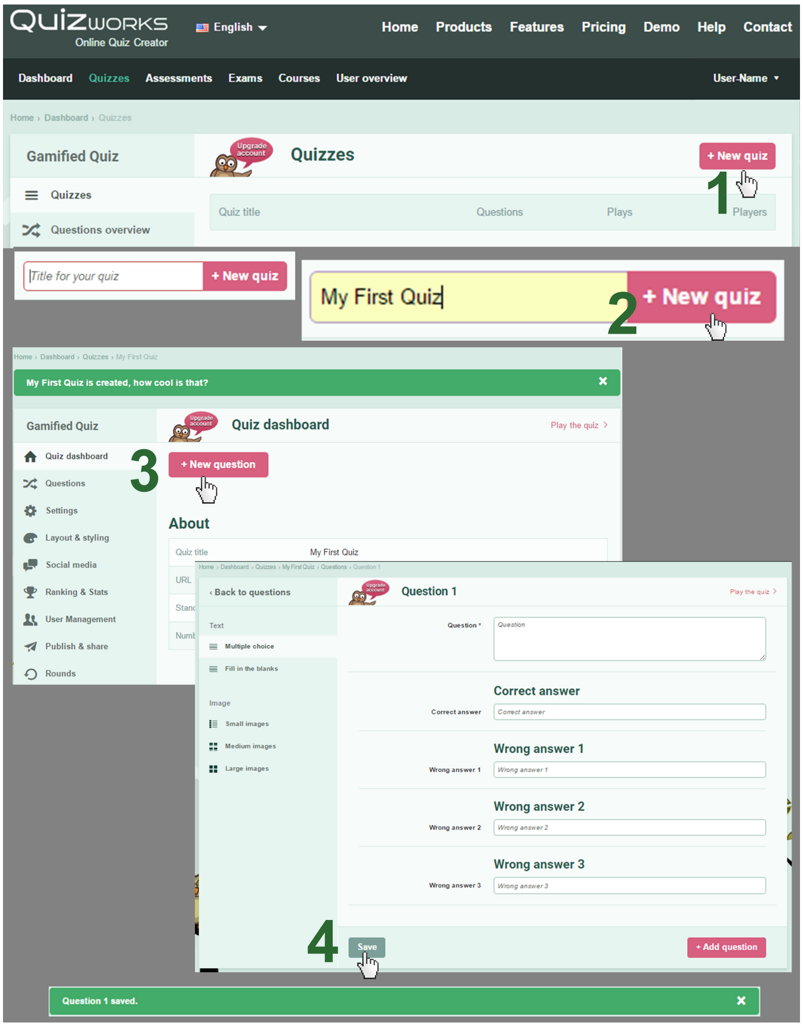
Researching similar applications is an important step before any development takes place. In this chapter, three on-line quiz generators (***QuizWorks, QZZR and QuizBean***) are detailed analysed. A short but full-length scenario was created for each of these systems, in order to present the flow of their primary use cases.

1.2 QuizWorks

"Playing and creating quizzes has never been more easy & fun!"

QuizWorks[5] is an online quiz creator developed by the homonym company (*ref) . It offers four different tools for creating Quizzes, Assessments, Exams and Courses and also provides numerous options to customize each product. Besides the many kinds of products, QuizWorks also offers lots of question types to choose from and advanced ranking & statistics options. The user can create, play, share quizzes or any other product created and view results. The system is equipped with a powerful help section which includes a search engine, a live chat with the help center and a frequently asked questions list that covers instructions and explanations related to every segment of the system.

In relation to user management, QuizWorks handles two kinds of users. The logged-in users and the "anonymous" users. The privileges of an unregistered/non-logged user are limited in QuizWorks. The user can have access to information regarding the system and even play quizzes created by others but registration is required for creating a product. Quiz, exam, assessment or course creation and publication is a privilege of a registered and logged-in user only.



My First Quiz

My First Quiz is created, how cool is that?

Quiz title: My First Quiz

Question 1 saved.

Illustration 1: QuizWorks - Quiz Creation

1.2.1 General Scenario of Use

Quiz Creation

The picture on the left illustrates the steps that a user has to follow in order to create a Quiz. For the purpose of this example, only one question will be added to the quiz.

The first thing that the user has to do is select the *+New Quiz* option, found under the Quizzes menu.

When this button is triggered, the system reveals an input field asking for a quiz title. After entering a title, the user must press once again the *+New Quiz* button (step 2).

In the upper part of the screen , the system shows a message informing the user that the quiz has been created. Now the user's screen is the Quiz dashboard section which among all includes a button for adding a question. The user shall select this option to go on (step 3).

A quiz can have three types of questions: Multiple choice, Fill the Blanks and Images. This example shows the creation of a multiple choice type of question. The user is expected to fill the given form with the question's title, one correct answer and one to three wrong answers. The user can keep adding questions by selecting the *+Add Question* option on the right bottom of the form. After question handling, the user can select the *Save* button to submit changes (step 4) and the system shows an informative message of the success.

1.3 QZZR

“Create, embed, and share your own quizzes”

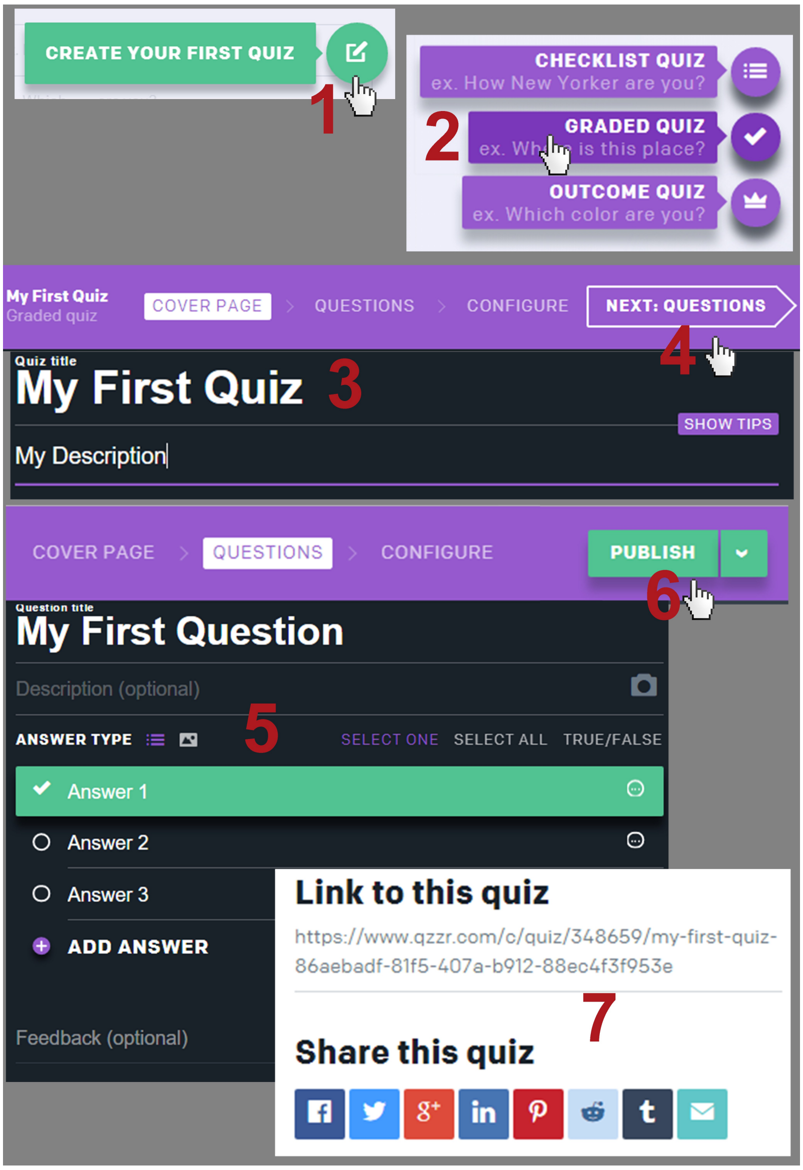
QZZR[6] is an on-line quiz tool created by Movement team. QZZR offers three kinds of quizzes labelled as Checklist Quiz, Graded Quiz and Outcome Quiz.

The Outcome Quiz is a kind of a personality test (ex. Which movie character are you?). The creator decides several *outcomes* and each *answer* of each question is connected to one of the outcomes. As the player chooses answers to the questions, the result is calculated by the system concluding to the final outcome at the end of the quiz. Similarly, the Checklist Quiz is a kind of personality test, the result of which is usually a percentage (ex. How much of a Harry Potter geek are you?). The Graded Quiz is a typical kind of quiz. The creator can add several questions with correct and wrong answers. The result of a player is shown as a percentage and as a number of questions that the player got right.

A registered user can create, edit, delete and share quizzes. Anonymous users can play quizzes created and shared by others but not create themselves.

1.3.1 General Scenario of Use

Create Graded Quiz



Quiz title: My First Quiz

My Description

Question title: My First Question

Answer 1

Answer 2

Answer 3

https://www.qzzr.com/c/quiz/348659/my-first-quiz-86aebadf-81f5-407a-b912-88ec4f3f953e

Illustration 2: QZZR - Create Graded Quiz

This scenario describes the creation of a Graded Quiz. As mentioned previously, only a registered user can create a quiz. For the purpose of this example we assume that the user has already created an account and the current screen is the dashboard screen.

For a new user, the dashboard page contains nothing but a button that reads *Create Your First Quiz*. The user can select this option to start the creation of a quiz (step 1).

When the above option is clicked, the system shows a list of options and asks the user to choose the type of the quiz about to be created. We assume that the user chooses the second option Graded Quiz (step 2).

After the uses selects the quiz type, the system asks for a the quiz' title. In the current screen there are more optional information to be given, like a description of the quiz and more that are not shown in the picture. The title given in the example is My First Quiz (step 3).

A strange thing about this system is the absence of a save button. Instead , the whole creation is done with the steps Cover Page → Questions → Configure where the user can go back and forth. So when the user is done entering the title, he/she can press the *Questions* button to continue or the *Next: Questions* option (step 4).

By now the system already asks for the title and the type of the first question. This kind of quiz has three options for question: multiple choice with one correct answer (*Select One*), multiple choice with multiple correct answers (*Select All*) and *True or False.* In this example only one question of type *Select One* is created. Once all the question's information is given, the user can continue. If the user selects the Next button (by now it says *Next: Configure*), the styling options will appear. In this example this step is skipped. Instead, the user will publish the quiz by pressing the *Publish* option (step 6).

When a quiz is published, the system provides a link of the quiz-play environment among other sharing options (step 7).

1.4 Quizbean

"A Quiz Creator to Quickly Assess Your Students."

Quizbean[7] is a powerful, online quiz generator tool, built by Bluehouse Group. Unlike the applications analyzed previously, Quizbean focuses excursively on the educational process of teaching and testing though quizzes. A registered user can create, play and share quizzes, copy quizzes from other users, create classrooms and add students to them, and join other teachers' classrooms.

1.4.1 User Management

Quizbean provides two types of registration, one for *teachers* and one for *students.* By creating a default account, the user is entitled to the role of a Teacher and the system provides a *teacher code* which works as a unique identifier among teachers. Every Teacher has the ability to create and share *quizzes,* to create *classrooms* and invite students to join them, and lastly, to validate results of the quizzes that has sent to students. The role of a teacher is not absolute, meaning that a teacher-user can also be a student who takes quizzes and is part of one or more classrooms created by other teachers.

Quizbean also offers a registration type specifically for Students. The required information for a student sign-up include an extra field, the *teacher's code* which was mentioned above. Once the student account is created, the teacher-user (who matches the code) can add the student-user to classrooms and send quizzes to the latter. A second way for student registration is through the teacher. A logged teacher can create multiple accounts for his/her students and add each student at a classroom during account creation. The capabilities of a student-user are limited in comparison to the teacher-user. A student can take quizzes sent by teachers, create quizzes and view his/her results. But a student cannot create a classroom or accounts for other students.

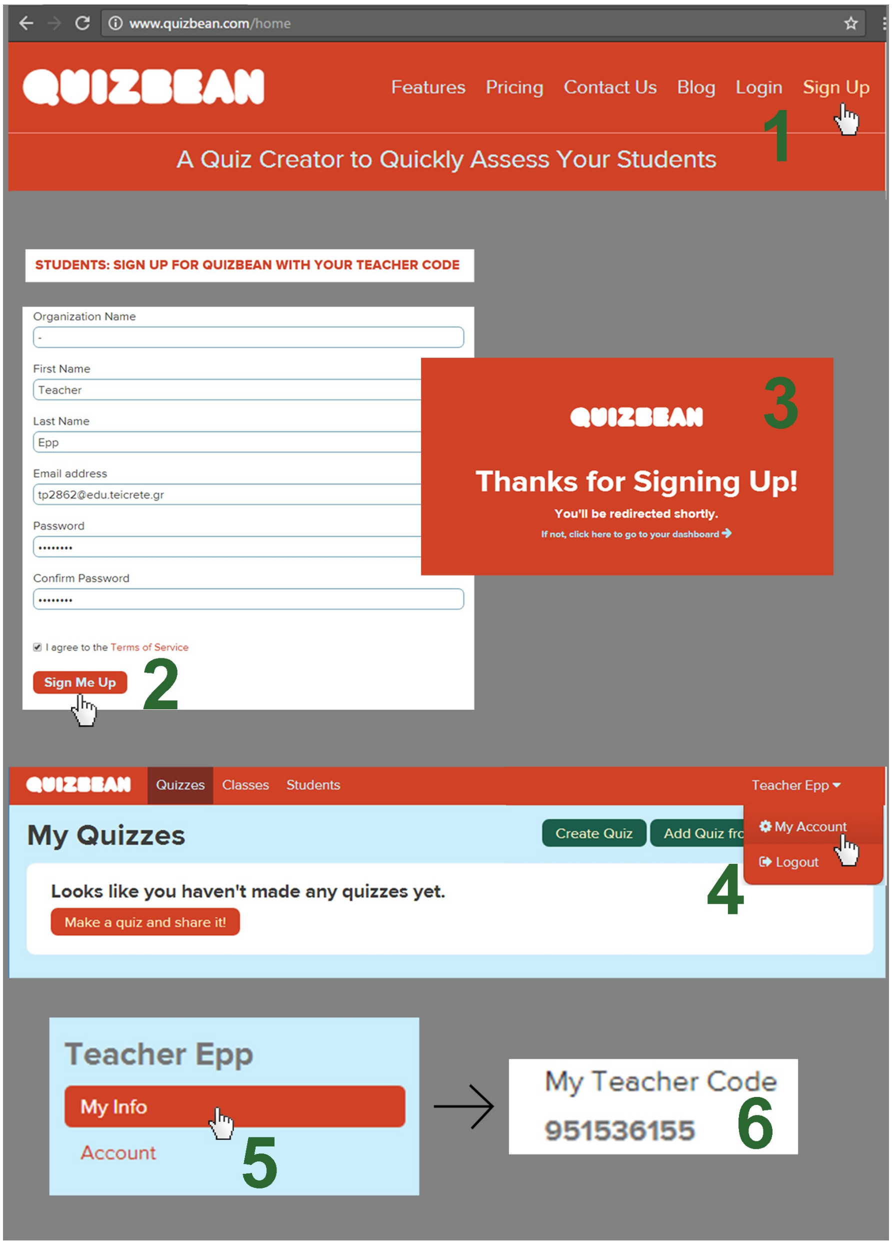
(…) logged and non-logged in

1.4.2 General Scenario of Use

Let's suppose that a teacher wants to use Quizbean system in order to send a quiz to one of his/her students. For the purpose of this example, both users will register to Quizbean by visiting the sing-up page and each has to submit the suitable form for the matching role (teacher or student). The teacher will create a quiz and send it to the student. Following that, the student will take this quiz and finally the teacher will view the results.

The following section contains a brief description of some scenarios in this order: Teacher Registration, Student Registration, Quiz creation by Teacher & Sent Invitation to Student and Student takes teacher's Quiz. Each example scenario uses real-time data and each scenario step is represented with screenshots of the actual system.

Teacher Registration



STUDENTS: SIGN UP FOR QUIZBEAN WITH YOUR TEACHER CODE

Organization Name: -

First Name: Teacher

Last Name: Epp

Email address: tp2862@edu.teicrete.gr

Password: ••••••••

Confirm Password: ••••••••

☑ I agree to the Terms of Service

Thanks for Signing Up!

You'll be redirected shortly.

If not, click here to go to your dashboard

Looks like you haven't made any quizzes yet.

My Teacher Code

951536155

Illustration 3: QuizBean - Teacher registration

The picture on the left is a screenshot representation of the scenario “Teacher Registration”, extended to the point that the user can find the *teacher's code*.

As the first steps shows, when the user visits the Quizbean page, the *Sing Up* option can be found in the top right corner. By selecting this option, the registration environment appears containing a teacher's registration form (default) and a link to student's sing-up.

Since the current scenario relates to a teacher, the user will fill the current form with the required information and press *Sing Me Up* (step 2).

Presuming the teacher has entered valid information on the first try, the registration is successful and the system shows a welcoming message before redirecting to the dashboard screen (step 3).

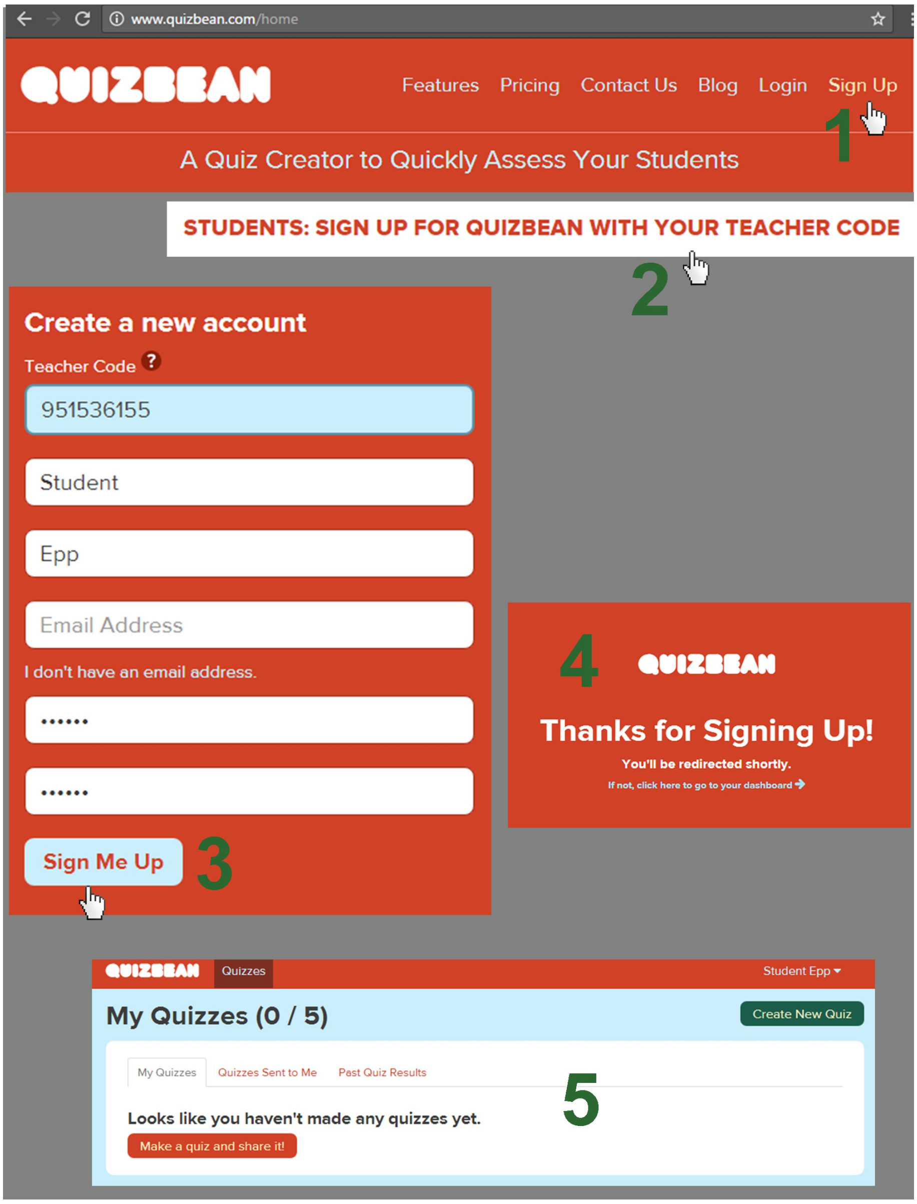
Now on the top right corner of the main menu, the teacher can find the *My Account* option under the user-name’s drop down menu (step 4).

By clicking this button, the teacher can access private information under the *My Info* option (step 5).

The information shown now include the teacher's full name, e-mail address and most importantly for this scenario, the *teacher's code* (step 6).

Teacher account creation must happen prior to student registration so that the teacher can provide his/her student with this code. Now that the code has been acquired, let's see how the student can register.

Student Registration



A Quiz Creator to Quickly Assess Your Students

STUDENTS: SIGN UP FOR QUIZBEAN WITH YOUR TEACHER CODE

Create a new account

Teacher Code: 951536155

Student

Epp

Email Address

I don't have an email address.

••••••

••••••

Thanks for Signing Up!

You'll be redirected shortly.

If not, click here to go to your dashboard

My Quizzes (0 / 5)

Looks like you haven't made any quizzes yet.

Illustration 4: QuizBean - Student Registration

The picture on the left represents the scenario “Student Sign Up”.

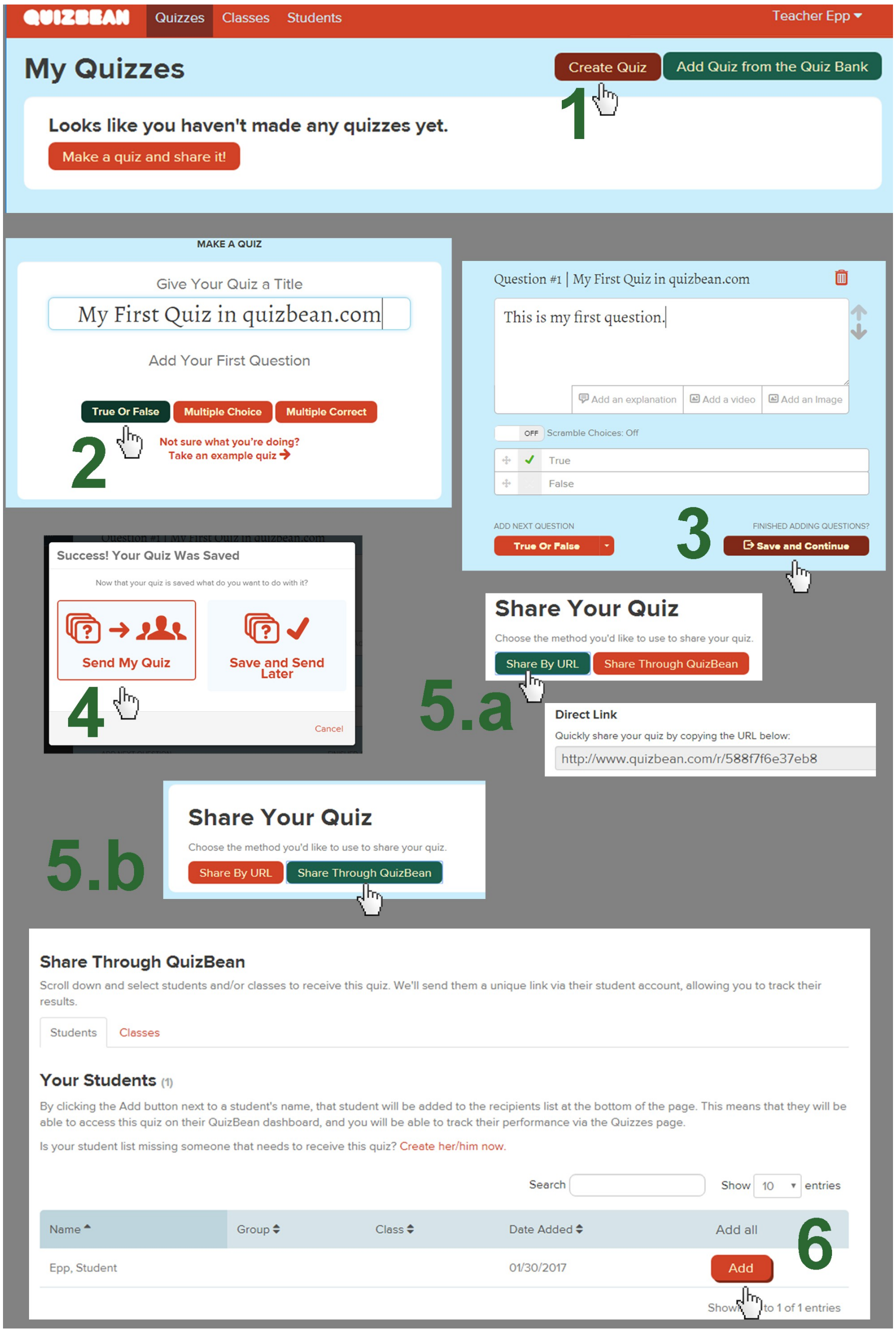
The first step is the same with the example analysed previously. The user can visit the Quizbean web-page and select the Sign Up option.

Unlike the teacher-sign up, the student will skip the registration form shown in the current screen and select the link *Students: Sign Up for Quizbean with your Teacher Code* (step 2). By clicking this option the system shows the special form for student registration.

Among the required information that the student is asked to enter, there is the teacher's code. The input must match a teacher-user in order for the account to be created. In this case, we assume that the teacher from the above scenario has provided the current student with the code. After filling the form, the student can user can submit it by selecting the *Sign Me Up* option (step 3).

We assume that the user has entered valid data during the registration process. Now that the student account has been created, the systems welcomes the student with a message (step 4). Finally, the student can redirect to the dashboard or the *Quizzes* page.

Quiz Creation by Teacher & Sent Invitation to Student



My Quizzes

Looks like you haven't made any quizzes yet.

MAKE A QUIZ

Give Your Quiz a Title

My First Quiz in quizbean.com

Add Your First Question

Question #1 | My First Quiz in quizbean.com

This is my first question.

Scramble Choices: Off

✓ True

False

Success! Your Quiz Was Saved

Now that your quiz is saved what do you want to do with it?

Share Your Quiz

Choose the method you'd like to use to share your quiz.

Direct Link

Quickly share your quiz by copying the URL below:

http://www.quizbean.com/r/588f7f6e37eb8

Share Your Quiz

Choose the method you'd like to use to share your quiz.

Share Through QuizBean

Scroll down and select students and/or classes to receive this quiz. We'll send them a unique link via their student account, allowing you to track their results.

Your Students (1)

By clicking the Add button next to a student's name, that student will be added to the recipients list at the bottom of the page. This means that they will be able to access this quiz on their QuizBean dashboard, and you will be able to track their performance via the Quizzes page.

Is your student list missing someone that needs to receive this quiz? Create her/him now.

Search Show 10 entries

Name	Group	Class	Date Added	Add all
Epp, Student			01/30/2017	Add

Show to 1 of 1 entries

Illustration 5: QuizBean - Quiz Creation by Teacher & Sent Invitation to Student

The picture on the left shows the steps for the creation of a quiz by a teacher-user. For the purpose of this example, only one question of type *True or False* will be added to the quiz. After the successful creation of the quiz the user will share it with the student.

In *Quizzes* page the teacher can find the *Create Quiz* option, located right under the main menu. When this button is triggered, the system reveals the quiz creation environment (step 1).

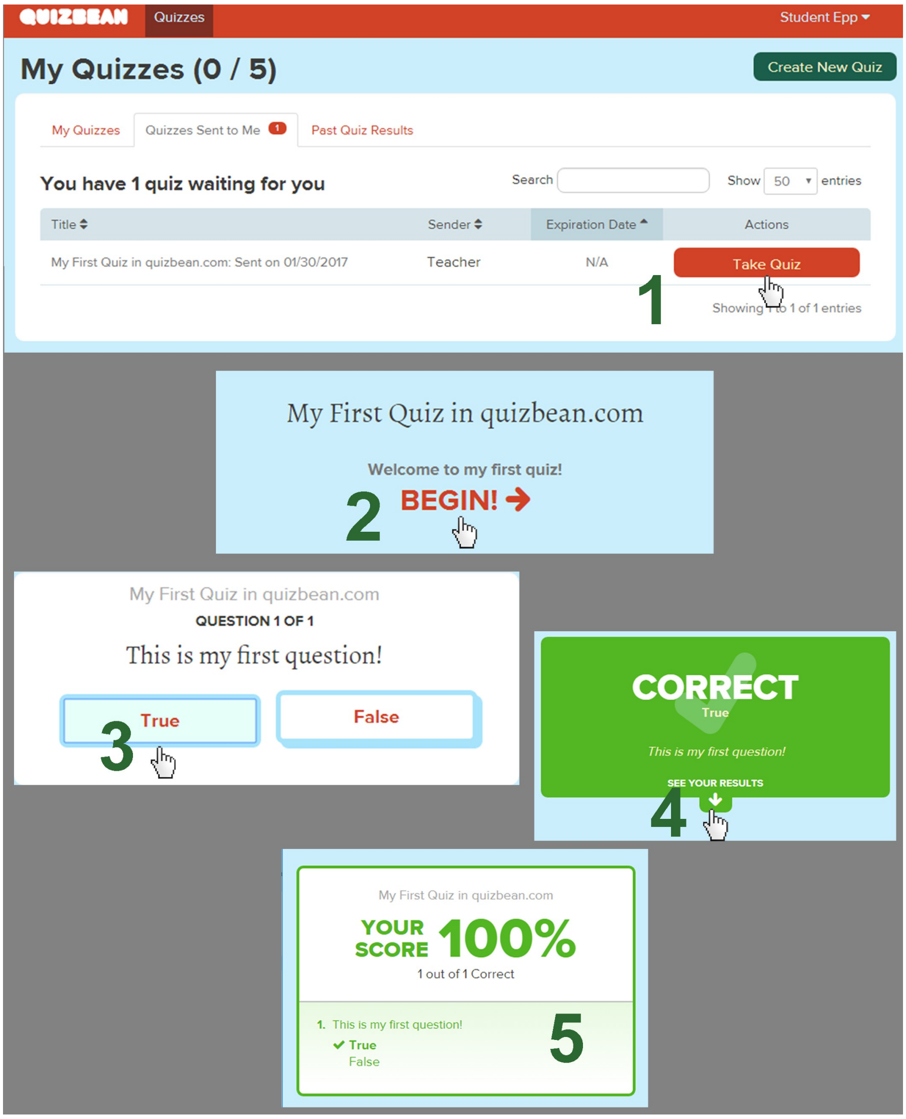
The given form requires only a quiz Title. Procedure can continue by selecting the type of the first question (step 2).

In the current screen, the teacher can enter the question's title and settings (picture, video, shuffle answers option etc.). To add another question, the user can select a question type from the drop-down menu located on the bottom left corner. Since this is a short example, we presume that the user selected the *Save and Continue* option which is on the bottom right (step 3).

Along with the message of successful quiz creation, the system provides the sharing option *Send My Quiz* (step 4).

When the teacher selects the above-mentioned button, the sharing settings screen appears. It contains the option of *Share by URL* (step 5.a) by which the user can send the link of the quiz to any student or other user. This environment also contains the Share Through Quizbean option (step 5.b). When the teacher chooses this, the system shows a list of all the students of the logged teacher. In this case there is only one student. By pressing the *Add* button (step 6) of this student, an invitation will be sent to the latter's account.

Student takes teacher's Quiz


QUIZBEAN
Quizzes
Student Epp
My Quizzes (0 / 5)
Create New Quiz
My Quizzes
Quizzes Sent to Me 1
Past Quiz Results
You have 1 quiz waiting for you
Search
Show 50 entries
1
Showing 1 to 1 of 1 entries
My First Quiz in quizbean.com
Welcome to my first quiz!
2
BEGIN! →
My First Quiz in quizbean.com
QUESTION 1 OF 1
This is my first question!
3
True
False
CORRECT
True
This is my first question!
SEE YOUR RESULTS
4
My First Quiz in quizbean.com
YOUR SCORE 100%
1 out of 1 Correct
1. This is my first question!
✓ True
False
5

Title	Sender	Expiration Date	Actions
My First Quiz in quizbean.com: Sent on 01/30/2017	Teacher	N/A	Take Quiz

Illustration 6: QuizBean - Student takes teacher's Quiz

This scenario describes how the student-user can take a quiz sent from the teacher-user. We assume that the above scenario has been completed successfully and that the student is logged-in and viewing the page *My Quizzes*, as the picture on the right shows

Under the tab *Quizzes Sent to Me*, the student can find a list of all invitation-quizzes. In this case there is only one. By clicking the button *Take Quiz* the system redirects to the game environment (step 1).

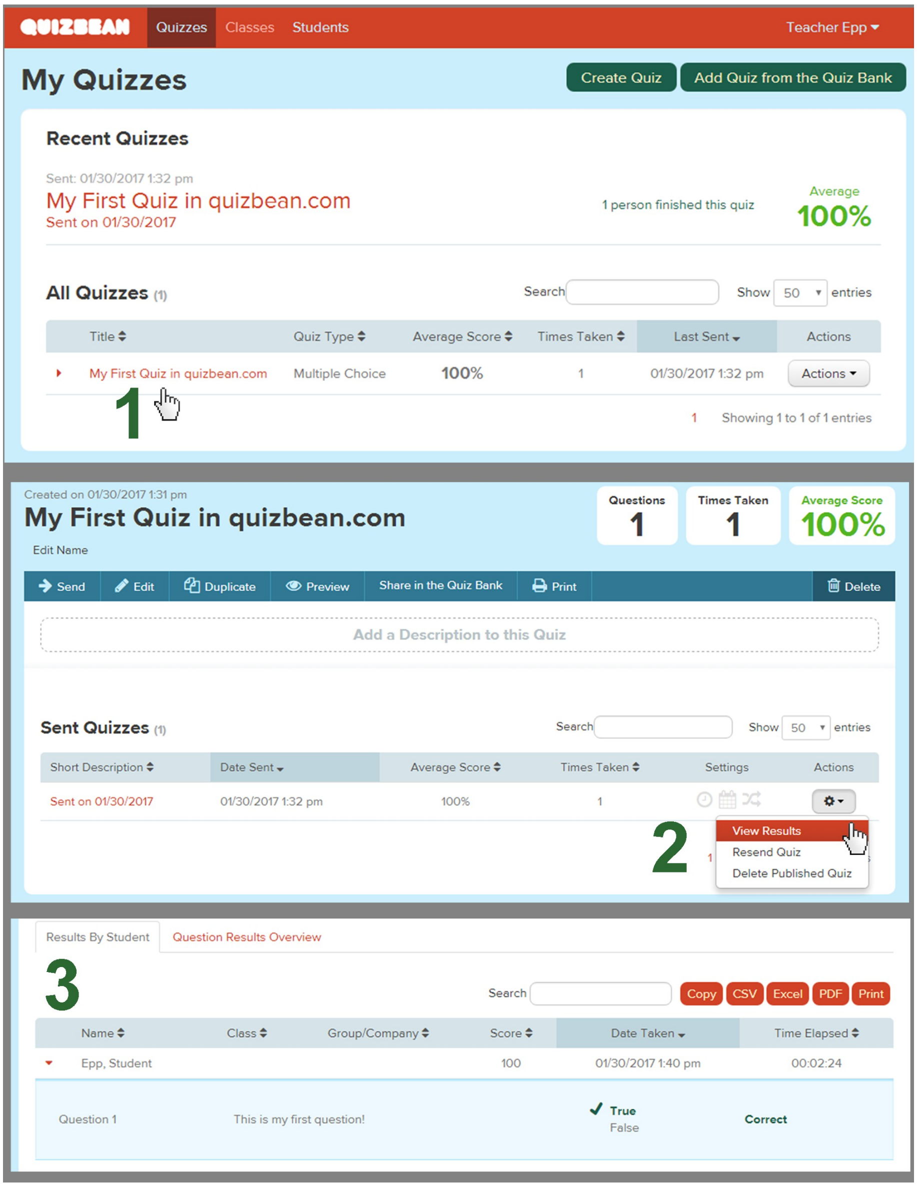
Before the quiz begins, the user is sees a welcoming screen that contains the quiz title, a message written by the teacher-user and a Begin button that triggers the beginning of the game. In order to continue, the user must click this button (step 2).

Questions during play time are presented one at a time. Each question panel contains the quiz title, the question's title, the index of the current question and the amount of total questions. According to the type of the current question there are also suitable options that the player can choose. In this example, the question type is *True or False* and we assume that the student hit the button *True* (step 3).

Quizbean only supports instant evaluation of each answer so after step 3, the system shows a message pointing out whether the answer is correct or not. In this case the chosen answer is correct. Since this is also the one and only question, the system provides the option *See your Results* that the student can select (step 4).

In the last step the student can see the score and also review every given answer. In this screen all questions are listed in the same screen in the order that they were taken. In this example the list contains only one question that was answered correctly.

Score Validation by Teacher



My Quizzes

Recent Quizzes

Sent: 01/30/2017 1:32 pm
My First Quiz in quizbean.com
Sent on 01/30/2017

1 person finished this quiz

Average 100%

All Quizzes (1)

Search Show 50 entries

Title	Quiz Type	Average Score	Times Taken	Last Sent	Actions
My First Quiz in quizbean.com	Multiple Choice	100%	1	01/30/2017 1:32 pm	Actions

1 Showing 1 to 1 of 1 entries

Created on 01/30/2017 1:31 pm
My First Quiz in quizbean.com
Edit Name

Questions 1 | Times Taken 1 | Average Score 100%

Sent Quizzes (1)

Search Show 50 entries

Short Description	Date Sent	Average Score	Times Taken	Settings	Actions
Sent on 01/30/2017	01/30/2017 1:32 pm	100%	1		

Search

Name	Class	Group/Company	Score	Date Taken	Time Elapsed
Epp, Student			100	01/30/2017 1:40 pm	00:02:24

Question 1 This is my first question! ✓ True False Correct

Illustration 7: Quizbean - Score Validation by Teacher

In this scenario we'll see how the teacher can view the student's results on the "invitation" quiz.

We assume that the teacher is logged-in and the current screen is the My Quizzes. Under the section All Quizzes, there is a list that shows the only quiz created so far. By selecting the link with the title of the quiz, the teacher is being redirected to the page of that particular quiz (step 1).

This page contains all information and options related to the current quiz. Under the drop-down menu *Actions*, the teacher can find and select the option *View Results* (step 2).

In this screen, the teacher can select a player from the list and each question along with the player's chosen answers will appear (step 3).

	onlinequizcreator.com	**qzzr.com**	**quizbean.com**
System Features			
Language Selection	Yes	No	No
Styling Options	Yes	Yes	Yes
Unlimited Quizzes	Yes	Yes	Yes
Mandatory Registration for Creators	Yes	Yes	No
Mandatory Registration for players	No	No	No
Exam	Yes	No	No
Course	Yes	No	No
Quiz	Yes	Yes	Yes
Rating Quiz	No	No	No
Time limit per Quiz	Yes	No	Yes
Instant Feedback / Instant score	Yes	Yes	Yes
Shuffle Questions	Yes	No	Yes
Shuffle Answers	Yes	Yes	
Expiration date	Yes	No	Yes
Create Quiz from bank of questions	No	No	Yes
Prefixed Quizzes	No	No	Yes
Question Types			
Multiple Choice	Yes	Yes	Yes
Free Text	Yes	No	No
Fill the Blank	Yes	No	No
Matching	No	No	No
Rate	No	No	No
True or False	Yes	Yes	Yes
Image	Yes	Yes	Yes
Video/Audio	Yes	No	Yes

2. Technology Involved

A web application[8] is a program that a user can access over a network via browser. Web apps, unlike websites, are more action-oriented rather than information-oriented. Usually the user must register/log in order to reach full capabilities of the application and store, retriveve, alter or delete data that corresponds to the logged user's account. The application part of such programs is not stored locally in the user's device like desktop software but in a remote server which makes offline functionality limited or non-existent. Despite this drawback, there are many advantages of web apps that surpass desktop software. For example, a web application doesn't require installation and as an extend, the app's version is always up to date. Also, the users can have access to their personal account data by logging into the system through any device that has a web browser.

The architecture of a web application can be separated into multiple layers that are also called *tiers*. Each tier has different responsibilities to handle and require different technologies and skills to develop. Complicated applications may have more than three tiers but the current project, like most web applications, follows the *3-Tier[9]* approach which has three, well-distinguished from each other layers. The first tier is the *Presentation Layer* which includes the logic associated with the application's user interface. Another layer is the *Data Access Layer* which, as the its name states, is a database which stores all the information that is needed. And last is the *Business Layer* which contains the core computations such as data flow handling, validation of requests and result delivery. Basically it is responsible for every task of the application and it's also the connector which allows communication between the other two layers. The image below represents the architecture and the communication of the layers.

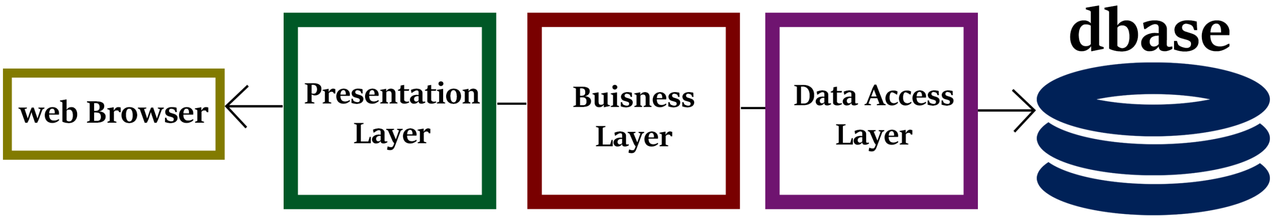


Illustration 8: 3-Tier Architecture

The implementation process of a web application consists of two main sections: the front-end development and the the back-end. By 'front end' part of the application we mean everything that the end user sees and interacts with. In other terms, the application's *client side* or, as mentioned above, the application's *Presentation layer.* This includes of course the design but also the direct functionality of the web page (e.g trigger of a drop down button). Technologies that are commonly used in front-end development are HTML/XHTML, CSS, Bootstrap, Javascript and jQuery.

As for the back end part, it consists of the development of the logic behind the Business and Data Access layers. Suitable languages for back-end development are PHP, Java, .Net, Python and others, and the most popular database systems are MySQL and PostgreSQL.

2.1 Presentation Layer Technologies

Bootstrap[10] is an open-source framework that provides user interface components for web site/application development. An example of what Bootstrap offered to this web application is the responsive design. This means that the layout dynamically adjusts to the user's device screen size without loosing any functionality. Other cases of Bootstrap usage in this project is the style of the buttons', the modal dialogs, the menu, the glyphicons (e.g Trash icon in delete buttons, Pencil icon in edit buttons e.t.c) and many others.

Most of the style classes used from Bootstrap were customized with CSS. CSS (Cascading Style Sheets) is a language that helps the developer to apply or alter the style of HTML/XHTML elements by changing their dimension, color, fonts and other parameters. Apart from a CSS file created for this specific project, the free animated.css[11] (*ref) was also included and used .

PrimeFaces[12] is an open-source user interface component suite for JavaServer Faces based applications. It provides a library of custom styled UI components, built-in Ajax, mobile features and much more.

2.2 Business Layer Technologies

This project was developed using Java Platform, Enterprise Edition, commonly known as Java EE[13] is a computing platform designed to support the development of enterprise applications. Java EE is built on top of Java SE (Standard Edition) therefore they use the same syntax and utilities (such as data structures). The difference between them is that Java EE extends the capability of the Standard Edition in order to achieve the requirements of complex and multi-tiered network systems. By simplifying the definitions we could say that Java EE is a framework designed for the development of business level web applications.

2.3 Data Access Layer Technologies

In object oriented programming a class is the 'blueprint' of it's objects and the state of each object is defined through its instance variables. Often a class is written in a way so that its instance will represent an object of the physical world and usually this information needs to be saved in a database.

Object-relational mapping (ORM) is a concept of converting objects into types that can be persisted in a relational database and the other way around. With ORM a *plain old java class* is called an *Entity.* Each entity can represent a table in a relational database, each variable of the class will be a table element and each instance of the class will be a row of information in that table. In this way, object references in the code are linked to instances from the database.

In order to implement the ORM technique there is a need of a framework that provides the 'translation' between a java class and a database table. This project uses the Hibernate[14] ORM framework. Hibernate hides the complexity of SQL queries by providing functions to persist, retrieve, update and delete information in the database. It also enables the definition of associations between tables and entities (mapping).

By using Hibernate, the tables and contents of the database were automatically created by deploying the project. The database itself though, has to be created manually. The database for this project was created through phpMyAdmin[15], an open-source tool that allows developers to handle MySQL[16] databases.

3. The MyQuizDB System

3.1 Our Approach

3.1.1 System Development Methodology

The software development methodology that the project will follow is decided prior to any implementation. It is the act of breaking down the development work into well-defined stages. There are many software process approaches, such as the Waterfall model[17] and the Prototyping Build Model (*ref). According to the waterfall model there are five stages that should be followed by this order: Requirement specification → Design → Implementation → Verification → Maintenance. In prototype model every result leads to a built which is evaluated and after re-defined and re-built according to its weaknesses and needs. This 'circle' of creating, testing and re-building breaks when the built matches the desired product.

This project used the Incremental Build method (*ref) which is considered to be a combination of the two models mentioned above. According to this method, the product is designed, implemented and tested incrementally until the product is finished.

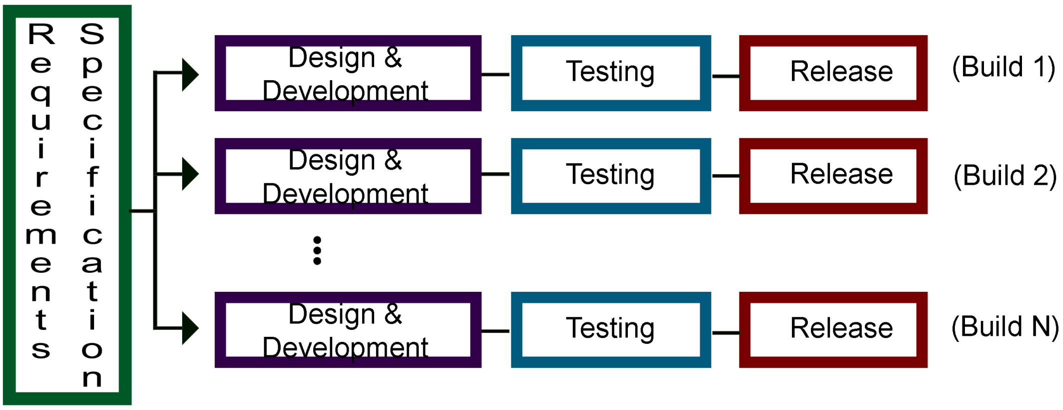


Illustration 9: System Development Methodology

3.2 System Analysis & Design

3.2.1 Application Requirements

In software engineering, requirement specification is a comprehensive description of what the desired software product must be able to do. All functions and capabilities of the system are written down in great detail and with a strict language that doesn't allow ambiguities. In that way, development time is minimised and the final product can be easily validated according to the original plan.

Requirements are divided into two categories: the functional and non-functional requirements. Non-functional requirements relate to system performance characteristics such as portability, security, efficiency, reliability and so on. On the other hand, functional requirements specify the behavior of the system (task handling, functions, services).

3.2.1.1 Non functional requirements

Usability

- Any user, regardless of experience status, must be able to use all the functions of the application after 1 hour of operating the system.
- Every enabled input must have an informative tool-tip message on mouse-over.

Portability

- The application must behave in the same way regardless of the browser that it is running on.
- The application must be functional in all screen sizes.

Security

- The system must not allow users to view or edit other users' private data.
- After inactivity of 60 minutes, the system will automatically invalidate the session of a logged user.

3.2.1.2 Functional Requirements

ID	Section	Name	Title
AC001	Account		The system must allow any user to login using a username and a password.
AC002	Account		The application must provide a registration environment.
BR001	Navigation		The first screen that a unauthenticated user must see is the login screen.

BR002	Navigation		The system must provide a redirection link towards the registration environment in the login screen.
-	-		The system must provide a “Continue as Guest” option in the login screen.
BR003	Navigation		A Guest User must be able to visit the registration environment through the main menu within one step.
BR004	Navigation		Any logged in user must be able to Log out within two steps, through the main menu.
BR005	Navigation		After logging in, the user will be redirected to the page *My Quizzes.*
BR006	Navigation		The navigation menu must always be visible to a logged user, excluding dialog -windows.
BR007	Navigation		In every dialog window there shall be a close/return button.
BR008	Navigation		The back button of a browser must redirect the user to the previous screen, taking into account user authorization restrictions.
GN001	General		The application shall provide a full functional page for each of the following sections: Quizzes, Questions, Invitation Quizzes, Pools, Results
GN002	General		After the creation of an application element, the system must inform the user of the success through a message.
GN003	General		After an invalid submission of a form, the system must indicate each input error with a message.
GN004			Every form of the system must indicate required input fields with a asterisk *.
GN005	General		When the user selects any delete button, the system will show a warning pop-up window with cancel and proceed options.
GN006	General	Input Texts characters Amount	Every text input must have the 255 maximum limit of acceptable characters with the exceptions of <u>Quiz Title</u> which must have maximum <u>500</u> characters and <u>the Answer & Question Title</u> which will accept up to <u>1000</u> characters.
QZ001	Page: *My Quizzes*	Quiz Info	The application shall provide *My Quizzes* page that contains all information and functions regarding the quizzes that the logged user has created. This information must be: 1)The total number of quizzes, 2)Title of each quiz, 3)Subject of each quiz, 4)Number of questions of each quiz.
QZ002	*Page: My Quizzes*	Quiz Creation	Through the *My Quizzes* page, the user must be able to open the quiz creation environment, with one step.
QZ003	*Page: My Quizzes*	Delete Quiz	The *My Quizzes* page shall provide a delete-quiz button in every panel that represents a quiz.
QZ004	*Page: My Quizzes*	Edit Quiz Settings	The *My Quizzes* page must provide an edit-quiz option that opens the edit quiz environment
QZ005	*Page: My*	Send Invitation	The *My Quizzes page* must provide a send invitation button in

	Quizzes		every panel that represents a quiz. The button must open a pop-up window that contains the send invitation environment
QZ006	*Page: My Quizzes*	Questions of a Quiz: Redirection	The *My Quizzes* page must provide a button that redirects to the Questions of each quiz page.
QZ007	*Page: My Quizzes*	Play Quiz: Redirection	The user must be able to go to the play quiz environment by selecting the *Play* button of each panel that represents a quiz.
QZ008	Quiz Creation	Quiz Creation Environment	The Quiz Creation environment must be in a pop-up window that contains a form with all the suitable input options. The input must be 1) Quiz Title: Text Area, 2)Quiz Subject: Text Area, 3) Publicity: Check-box, 4)Type: Radio buttons 5) Initialize points: Spinners. 6)Initialize Question comment :Text Area, 7)Allow shuffle questions :Check-box, 8)Allow reveal answer button during play-session.
SI001	Send Invitation for a Quiz		The send invitation environment must provide a selectable list with all the users of the system.
SI002	Send Invitation for a Quiz		The user must be able to search another user by inserting the desired user-name or e-mail address in the search field.
SI003	Send Invitation for a Quiz		The system must show a suitable message if the search doesn't match any results.
SI004	Send Invitation for a Quiz		The system shall not allow the user to re-sent the same invitation more than once.
SI005	Send Invitation for a Quiz		If an invitation has already been sent, the system will disable the selection of the receiver from the list, and will show an informative message.
SI006	Send Invitation for a Quiz		A user must be able to select multiple users to send an invitation.
SI007	Send Invitation for a Quiz		The user must be able to submit the invitation form by selecting *Send.*
SI008	Send Invitation for a Quiz		After sending an invitation, the system must inform the user of the success with a message.
SI009	Send Invitation for a Quiz		After sending an invitation, the window with the corresponding environment closes and the user returns to the *My Quizzes* page.
P001	Page: *Pools*		The application must provide a *Pools* page that contains all information and functions regarding the question pools of the logged user.
P002	Page: *Pools*		Through Pools page , the user must be able to open the

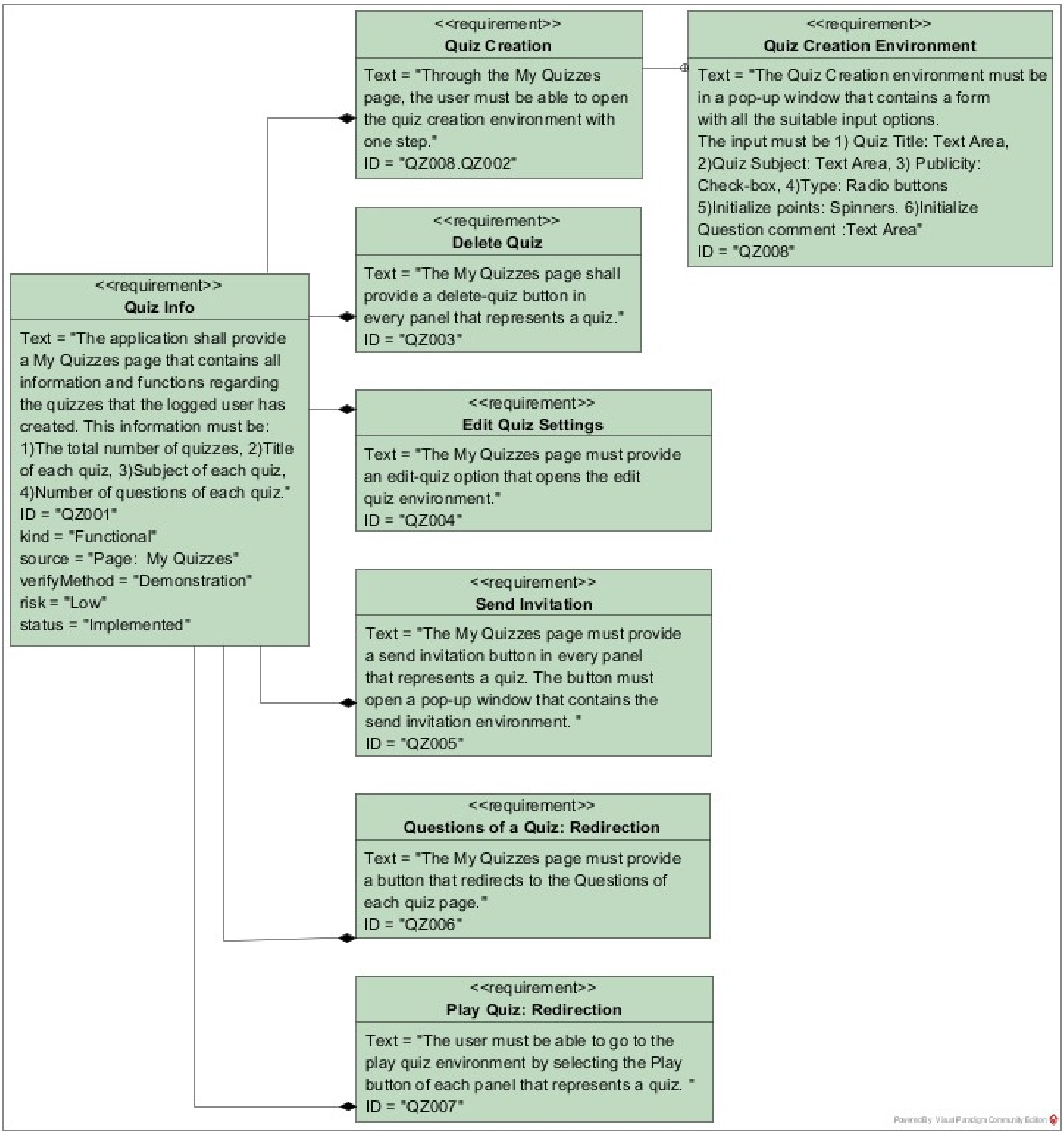
			environment for pool creation with one step.
P003	Page: *Pools*		Pools page must provide a delete-pool-button in each panel that represents a pool.
P004	Page: *Pools*		Pools page must provide an edit-pool settings-button in every panel that represents a pool, excluding “Orphan Question” pool if exists.
P005	Page: *Pools*		Through the *Questions* button of each pool, the user can be redirected to the questions page of a selected pool.
QS001	Menu/ Page *Questions*		The application shall provide a Questions page that contains all information related to the questions of quizzes and pools of the logged user.
QS002	*Page: Questions of a Quiz*		When the user is in the screen *Questions of a specific quiz,* the page shows only the questions that belong to that quiz.
QS003	*Page: Questions of a Quiz*		The *Questions of a specific quiz* screen must present the title of the current quiz as a link. This link shall open the edit quiz settings dialog.
QS004	*Page: Questions of a Pool*		When the user is in the screen *Questions of a specific pool,* the page shows only the questions that belong to that pool.
QS005	*Page: Questions of a Pool*		The *Questions of a specific pool* screen must present the title of the current pool as a link. This link shall open the edit pool settings dialog.
QS006	*Page: Questions*		Through page Questions, the user must be able to open the *Create new Question* dialog with one step.
QS007	*Page: Questions*		In the header of each panel that represents a question, the system must provide an edit question settings button that opens the corresponding dialog.
QS008	*Page: Questions*		In the header of each panel that represents a question, the system must provide a delete question option that opens the corresponding dialog.
QS009	*Page: Questions*		In the header of each panel that represents a question, the system must provide an add question to pool option that opens the corresponding dialog.
QS010	*Page: Questions*		In the header of each panel that represents a question, the system must provide a check-box option that defines the question as selected or deselected.
QS011	*Page: Questions*		The user must be able to manipulate multiple questions by selecting them from their check-boxes.
QS012	*Page: Questions*		The user must be able to select/deselect all of the shown questions of the page by checking/un-checking the *Select All* check-box
QS013	*Page: Questions*		Multiple manipulation options shall be those of *Delete Selected Questions, Add Selected to Quiz* and *Add Selected to Pool.*
QS014	*Page:*		Multiple manipulation options must be enabled if at least one

	Questions		of the questions listed in the page is selected.
QS015	*Page: Questions*		The user must be able to filter the view of the questions listed in the page by selecting a Quiz or a Pool from the given drop-down lists.
QS016	Question Creation	Question Creation Environment Features	The Question Creation environment must be in a pop-up window that contains a form with all the suitable input options. The standard options must be: 1) Question Title: Text Area, 2) Question Comment: Text Area, 3) Question Points: Spinner (according to reward type) 4)Answer's type: Radio buttons (with the exception of rate, see QS017) .
QS017	Question Creation	Question Creation Environment: Rating Quiz restrictions	A rating style quiz must not allow other kinds of player-answers other than rate.
QS018	Question Creation	Question Creation Environment: Answers' type	The radio-buttons' options of the answer's type (see QS016) must be 1) Free text, 2) Multiple choice, 3) Rating style, 4)Numeric input
QS019	Question Creation	Answer's type: Free text	When the *Free Text* option is selected, the system shall provide a read-only text area with the informative placeholder “The user will type an answer”.
QS020	Question Creation	Answer's type: Numeric	When the *Numeric* option is selected, the system shall provide two optional numeric input fields through which the user can define a minimum and maximum number restrictions to the player's answer.
QS021	Question Creation	Answer's type: Rate	When the *Rate* option is selected, the system shall provide an input spinner through which the user can define the amount of stars that the player will choose. * Restriction: The maximum amount of stars must be 10.
QS022	Question Creation	Answer's type: Rate & Quiz type Rate	When the rate type of answer comes from the type of the Quiz (Rating style), the amount of stars shall be already initialized from the quiz creation environment and shall not be able to change through the question creation phase.
QS023	Question Creation	Answer's type: Choice - Form Options	When the *Choice* option is selected, the system shall provide a form for adding multiple answers. This form must contain: 1)The answer's title: Text Area, 2) An “Add answer” option: button, 3)A 'set as correct' option: Check-box, 4)An 'input points' option: Spinner
IN001	Page: *Invitations*		The system must provide an *Invitations* page that shows all the information of the quizzes that the user has been invited to play. These information is 1) The title of each quiz, 2) The subject of each quiz, 3) The username of the invitation sender, 4)The number of questions of each quiz 5)The results of the quiz (in case it has been played)

IN002	Page: *Invitations*		The page must list in different panels the quizzes that have already been played by the logged user and those that await.
IN003	*Page: Invitations*		The user must be able to redirect to the Play Quiz environment by selecting the *Play* button of a quiz.
IN004	*Page: Invitations*		The header of each panel that represents a quiz must contain a *Delete Quiz* button.
VR001	Page: *View Results*		The system shall provide a View Results page that contains all the information of played Quizzes that the logged user is authorized to see. This information is
VR002	Page: *View Results*		The logged user is authorized to see the results of quizzes that: 1)The logged user has created and played 2)Other users have played thought invitation by the logged user
VR003	*Page: View Results*		The page must provide a selectable list with all the play sessions that the logged user is authorized to view. The information that the list will show is: 1)Quiz title, 2)Player username 3) Points earned, 4)Correct answers count, 5)Quiz type
VR004	*Page: View Results*		When a play session of the list is selected, the page must show all details related to that session. These details are 1)Title and subject of quiz of the selected play session, 2)Points earned, 3) Number of questions of quiz. 4)Every question and user answer of the quiz.
VR005	*Page: View Results*		The user must be able to browse through the questions of the selected play session by clicking the back and forward buttons.
VR006	*Page: View Results*		The user shall be able to view all details of current question. The standard details for all cases are: 1)Question title, 2) Player's chosen answer 3)Points earned of current question , 4)Player's chosen answer, 5) Validation of answer, 6)Correct answers
VR007		Validation of answers: Quiz with points	When the play-session that is currently validated has a reward type with points, the system must provide the following information and options next to each question's title: 1) An red 'X' or a green 'check' symbol according to weather the answer is considered right or wrong, 2)The points that were earned or the negative points that were lost 3)The option to declare the answer as correct (in case of free text or free numeric type of answer)
VR008	*Page: View Results*	Chosen Answer Validation: Multiple Choice with Points	When the play-session that is currently validated has a reward type with points and the current question's type is multiple choice, the system must provide extra information. This information is: 1)The correct chosen answers must be distinguished with a green background color, 2)The correct not chosen answers must be distinguished with a green outline, 3)The wrong chosen answers must be distinguished with a red

			background color.
VR009	*Page: View Results*	Chosen Answer Validation: Multiple Choice of Simple Questionnaire	In case that the Quiz is a *Simple Questionnaire* and the question that is viewed is multiple choice, the player's chose shall be distinguished by its yellow background color.
VR010	*Page: View Results*	Chosen Answer Validation: Free Text with Points	When the play-session that is currently validated has a reward type with points and the current question's type is Free Text, the system must provide a check-box option that set's the player's answer as correct.
PL001	*Play*	Info before Play	The system must inform the user of the Quiz that is about to be taken. This information about the quiz must be 1)The title, 2)The subject, 3) The points system
PL002	*Play*	Start button	The system must provide a *Start Quiz* button. When this button is triggered, the quiz starts.
PL003	*Play*	Question in Play	While the user takes a quiz, must be able to view 1)The index of the current question, 3)The total amount of questions, 4)The title of the current question, 5)The title of each answer 6)
PL004	*Play*	Functionality of Play:Select Answer	A user must be able to select an answer from the available list of answers by clicking on it.
PL005	*Play*	Functionality of Play:Browse	The user must be able to go to the next or previous question but selecting the corresponding buttons.
PL006	*Play*	Functionality of Play:Browse	If the quiz that is being played is *Public,* the user must be able to add each question in a Pool by selecting the *Add to Pool* button (plus icon).
PL007	*Play*	Submit Quiz	At any time, the user must be able to submit the quiz by selecting the button *Submit*.

3.2.1.3 Requirements Diagram of section 'My Quizzes page'



<<requirement>>
Quiz Creation

Text = "Through the My Quizzes page, the user must be able to open the quiz creation environment with one step."
ID = "QZ008.QZ002"

<<requirement>>
Quiz Creation Environment

Text = "The Quiz Creation environment must be in a pop-up window that contains a form with all the suitable input options.
The input must be 1) Quiz Title: Text Area, 2)Quiz Subject: Text Area, 3) Publicity: Check-box, 4)Type: Radio buttons 5)Initialize points: Spinners. 6)Initialize Question comment :Text Area"
ID = "QZ008"

<<requirement>>
Delete Quiz

Text = "The My Quizzes page shall provide a delete-quiz button in every panel that represents a quiz."
ID = "QZ003"

<<requirement>>
Quiz Info

Text = "The application shall provide a My Quizzes page that contains all information and functions regarding the quizzes that the logged user has created. This information must be: 1)The total number of quizzes, 2)Title of each quiz, 3)Subject of each quiz, 4)Number of questions of each quiz."
ID = "QZ001"
kind = "Functional"
source = "Page: My Quizzes"
verifyMethod = "Demonstration"
risk = "Low"
status = "Implemented"

<<requirement>>
Edit Quiz Settings

Text = "The My Quizzes page must provide an edit-quiz option that opens the edit quiz environment."
ID = "QZ004"

<<requirement>>
Send Invitation

Text = "The My Quizzes page must provide a send invitation button in every panel that represents a quiz. The button must open a pop-up window that contains the send invitation environment. "
ID = "QZ005"

<<requirement>>
Questions of a Quiz: Redirection

Text = "The My Quizzes page must provide a button that redirects to the Questions of each quiz page."
ID = "QZ006"

<<requirement>>
Play Quiz: Redirection

Text = "The user must be able to go to the play quiz environment by selecting the Play button of each panel that represents a quiz. "
ID = "QZ007"

Illustration 10: Requirements Diagram of section 'My Quizzes page'

3.2.2 Use Cases & Scenarios of Use

In software engineering, a use case scenario is defined as a series of interactions between the software product and the user (or even another system) which in this case is called an *actor*. Each scenario describes the steps that the actor shall take to accomplish a specific goal and the feedback that the system gives. Use cases and scenarios are used to reinforce software analysis process along with requirements specification. Usually, they are documented following a template and displayed though diagrams or even represented with mock-ups.

The Use cases and scenarios defined for this system are listed below:

1. Use case: **Creation**. Scenarios:
   - Create Account
   - Create Quiz
   - Create Pool
   - Create Question
   - Create Answer
2. Use case: **Delete**. Scenarios:
   - Delete a Quiz by its author
   - Delete Quiz from Invitations list
   - Delete Pool
   - Delete Question
   - Delete multiple Questions at once
   - Delete Answer
3. Use case: **Edit**. Scenarios:
   - Edit Quiz settings
   - Edit Pool settings
   - Edit Question's features
   - Edit Answer's features
4. Use Case : **Add**. Scenarios:
   - Add new question to Quiz
   - Add multiple questions to a Quiz
   - Add question to Pool
   - Add multiple questions to a Pool
5. Use case: **Copy**. Scenarios:
   - Copy question(s) from one quiz to another
   - Copy a question from an invitation-quiz to a pool during play-session
6. Use case: **Send**. Scenarios:
   - Send invitation for a quiz to one or more users
7. Use case: **Play**. Scenarios:

  - Play quiz
8. Use case: **View Results**. Scenarios:
  - View the results of a quiz a play-session
9. Use case: **Log In**. Scenarios:
  - Log to Account

3.2.2.1 Scenarios of use case : Creation

Create Account

Actors: Client

Preconditions: Client is registered in the system

Basic Flow of scenario

1. Selection of 'Register' link to redirect to the Register screen
2. Filling the registration form
3. Form submission
4. Account is created and user is logged in the system

Alternative Flow of scenario

4.a Input is not accepted and user re-enters the invalid info.

Detailed Description of Basic Flow of Events

1. Selection of 'Register' link to redirect to the Register screen

The first screen that a not logged user sees is the log in screen. In order to redirect to the register environment, the user must select the *Register* link.

2. Filling the registration form

The user must insert all the required information into the registration form. This information is a Username, an e-mail address, a Password and the password configuration field.

3. Form submission

After filling the form, the user must submit it by selecting the *Submit* button. By submitting the form, the system validates the user's input.

4. Account is created and user is logged in the system

If the given data is accepted, the system shows an welcoming message and the user is redirected in the *My Quizzes* page.

Detailed Description of Alternative Flow of Events

4.a Input is not accepted and user re-enters the invalid info.

If the given data is not accepted, the system shows informative error messages in the fields that need to be changed. The flow continues from Step 2: Form Submission

Create Quiz

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *My Quizzes* page

Basic Flow of scenario

1. Selection of +*New Quiz* button
2. Filling the required fields of quiz creation form
3. Filling the optional fields of quiz creation form
4. Submit form
5. Quiz has been created and system redirects to *Questions* page

Alternative Flow of scenario

5. a. Input is not accepted and system informs the user accordingly

Detailed Description of Basic Flow of Events

1. Selection of +*New Quiz* button

The user must trigger the appearance of the Quiz Creation Environment by selecting the +*New Quiz* button.

2. Filling the required fields of quiz creation form

The user must fill all the required information of the given form. This information is the title of the quiz and the reward type.

3. Filling the optional fields of quiz creation form

The user can fill the optional information which is the subject and publicity of the quiz and also the initialization of the points (according to the reward type that was chosen)

4. Submit form

By selecting the *Save* button, the form is submitted and the system validates the given input.

5. Quiz has been created and system redirects to *Questions* page

If the given data is accepted, the system informs the user with a message and redirects to the page *Questions of the quiz that was just created.*

Detailed Description of Alternative Flow of Events

5. a. Input is not accepted and system informs the user accordingly

If the data that the user entered is not correct, the system indicates the fields which need to be changed with a message. After correcting the information, the user must repeat form submission (step 4).

Create Pool

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *Pools* page

Basic Flow of scenario

1. Selection of +*New Pool* button
2. Enter the pool's title
3. Submit form
4. Pool has been created and client is redirected to the Questions page

Alternative Flow of scenario

4. a. The pool's title is invalid and the pool is not created

Detailed Description of Basic Flow Steps

1. Selection of +*New Pool* button

The user must select the +*New Pool* button in order to trigger the appearance of the Pool Creation Environment.

2. Enter the pool's title

The creation of a pool requires one and only field, the title of the pool which the user must enter.

3. Submit form

By pressing the *Save* button the form is submitted and the system validates the input.

4. Pool has been created and client is redirected to the Questions page

If the given pool's title is accepted, the system shows an informative message of the success and the user is redirected to the *Questions of the pool that was just created* page.

Detailed Description of Alternative Flow of events

4. a. The pool's title is invalid and the pool is not created

If the pool's title is not accepted, the system indicates the reason under the input field. The user must re-do the steps 2 and 3 until a pool is created.

Create Question

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *Question's* page

Includes: Scenario 'Create Answer'

Basic Flow of scenario

1. Selection of *+Add Question* button
2. Fill all the required fields of the form without changing the initialized type *Multiple Choice*
3. ***Scenario: Create Answer***
4. Fill the optional fields of the form
5. Submit form
6. Question is created

Alternative Flow of scenario

2. a. i. User changes question's type to *Free Text*

   a. ii. The flow will continue from Step 5 (Submit form)

2. b. i. User changes question's type to *Rate*

   b. ii. User enters amount of rating stars

   b. iii. The flow will continue from Step 5 (Submit form)

2. c. i. User changes question's type to *Numeric*

   c. ii. User fills optional Numeric Fields

2. d.. If Question belongs to Rating Type of Quiz the flow will continue from Step 5

6. a. Given input is not accepted.

Detailed Description of Basic Flow of Events

1. Selection of *+Add Question* button

In order to make the question's creation environment appear, the user must select the *+Add Question* button.

2. Fill all the required fields of the form without changing the initialized type *Multiple Choice*

The standard fields that the user is required to enter are the *title* and *type* of the question. The default type is the *Multiple Choice* so this option is initialized accordingly.

3. ***Scenario: Create Answer***

4. Fill the optional fields of the form

The standard optional filed is a *comment* or *hint* that will appear during play time.

5. Submit form

By selecting the *Save* option the user can submit the form. In this step the system validates the given data.

6. Question is created

If the user's input is valid, the question is created and the system shows an informative success message. The question's creation environment closes and the user returns to the Question's page.

Detailed description of Alternative flow of events

2. a. i. User changes question's type to *Free Text*

By selecting the Free Text option, the question doesn't have any other required input data in order to be created. A read-only text area will appear and the placeholder will inform the user that the *Player will write a free text answer.*

2. a. ii. The flow will continue from Step 5 (Submit form)

A free text type of question doesn't have any optional fields associated with the answer so the user can continue by just submitting the form.

2. b. i. User changes question's type to *Rate*

If the *Rate* option is selected, the system will display an extra required input field.

2. b. ii. User enters amount of rating stars

The user is required to enter the amount of stars that the player will be given to fill through a numeric input-spinner.

2. b. iii. The flow will continue from Step 5 (Submit form)

A *Rate* type of question doesn't have any optional fields associated with the answer so the user can continue by just submitting the form.

2. c. i. User changes question's type to *Numeric*

If the *Numeric* option is selected, the system will display two extra optional input fields.

2. c. ii. User fills optional Numeric Fields

A Numeric type of answer may have minimum and maximum limits that the user can specify through the given input fields.

2. d.. If Question belongs to Rating Type of Quiz the flow will continue from Step 5

In this case the amount of stars that the player will be given to fill is already indicated during quiz creation. The user can't alter this information from the question's creation environment. Since no additional information is required, the flow will continue from step 5 (Submit form).

6. Given input is not accepted

If the given data is not accepted, the system informs the user with messages. The user must correct the invalid information and continue from Step 5 (Submit form).

Create Answer

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the Question's edit/create dialog environment
3. The question's type is *Multiple Choice*

Extends: Scenario 'Create Question'

Basic Flow of scenario

1. Fill required information
2. Fill optional information
3. Select +*Add* option
4. User saves question

Detailed Description of Basic Flow of Events

1. Fill required information

User is required to fill the title of the answer

2. Fill optional information

The optional information that the user can give is the points of the particular answer (according to the reward system) though an input spinner. By selecting the *Correct* check-box, the user can define the current answer as a right choice.

3. Select +*Add* option

By selecting the this option, the answer is added to the question's answers list. The amount of answers is increasing in the user's screen which also displays a new panel that represents the answer that was just added.

4. User saves question

By selecting the Save button, the answer is saved as part of it's question.

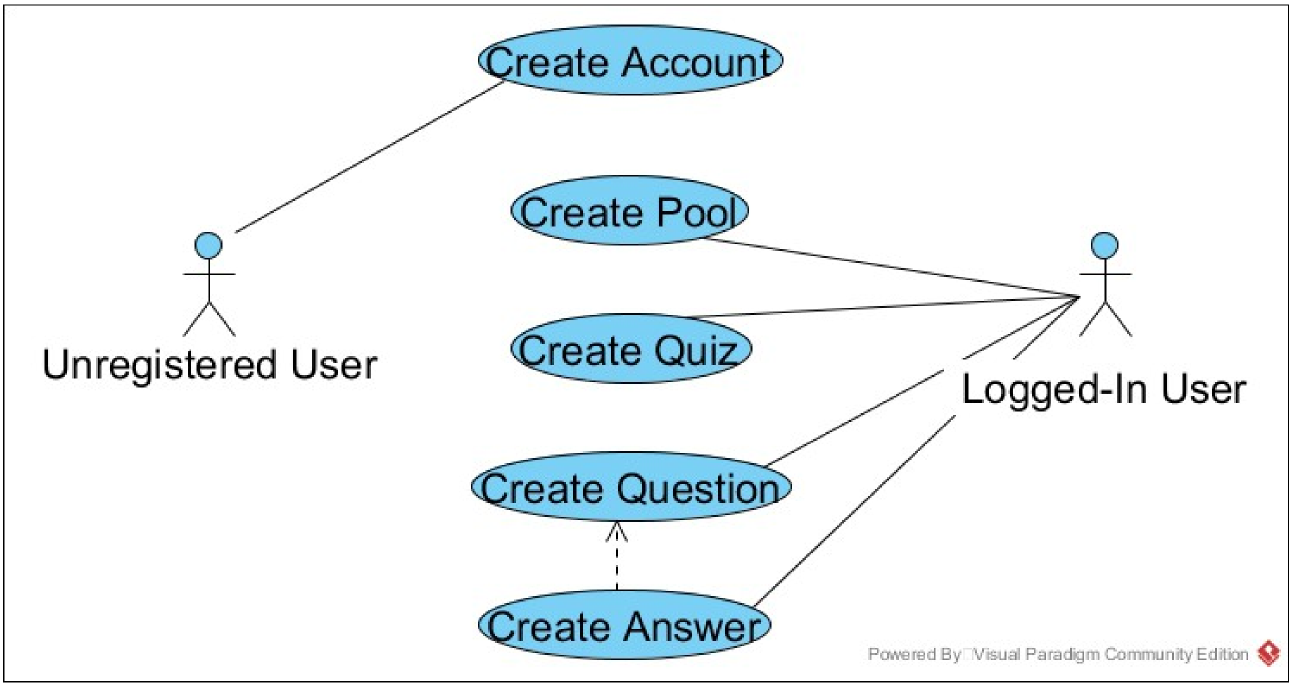


Illustration 11: Graphic Representation of Scenarios of use case : Creation

3.2.2.2 Scenarios of use case : Delete

Delete a Quiz by its author

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *My Quizzes* page

Flow of scenario

1. Select the trash-icon button of the panel that represents a quiz
2. Confirm delete
3. Quiz is deleted and system shows informative message

Detailed Description of Flow Events

1. Select the trash-icon button of the panel that represents a quiz

Each panel that represents a quiz contains a delete option in the form of a trash-icon. By selecting this option the system will pop-up a confirmation window.

2. Confirm delete

The user must confirm deleting the quiz by selecting the *Yes* option of the confirmation dialog.

3. Quiz is deleted and system shows informative message

In this step the system deletes the quiz and every element associated with it (questions, invitations, results). A message appears to inform the user of the successful delete.

Delete Quiz from Invitations list

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *Invitations* page

Flow of scenario

1. Select the trash-icon button of an invitation quiz
2. Confirm delete
3. Quiz is deleted and system shows informative message

Detailed Description of Flow Events

1. Select the trash-icon button of an invitation quiz

Each panel that represents an invitation quiz contains a delete option in the form of a trash-icon. By selecting this option the system will pop-up a confirmation window.

2. Confirm delete

The user must confirm deleting the quiz by selecting the *Yes* option of the confirmation dialog.

3. Quiz is deleted and system shows informative message

The system shows a message to inform the user of the successful action. By deleting an invitation quiz, the results of a possible play-session of the logged user are also deleted.

Delete Pool

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *Pools* page

Flow of scenario

1. Select the trash-icon button of a pool
2. Confirm delete

3. Pool is deleted and system shows informative message

Detailed Description of flow of events

1. Select the trash-icon button of a pool

Each panel that represents a pool contains a delete option in the form of a trash-icon. By selecting this option the system will pop-up a confirmation window.

2. Confirm delete

The user must confirm deleting the pool by selecting the *Yes* option of the confirmation dialog.

3. Pool is deleted and system shows informative message

The system shows a message to inform the user of the successful action. By deleting a pool, the questions that belong to that pool are also deleted. If the pool's questions also belong to a quiz, those questions will remain in that quiz.

Delete Question

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *Question's* page

Flow of scenario

1. Select the trash-icon button of a question
2. Confirm delete
3. Pool is deleted and system shows informative message

Detailed Description of flow of events

1. Select the trash-icon button of a question

Each panel that represents a question contains a delete option in the form of a trash-icon. By selecting this option the system will pop-up a confirmation window.

2. Confirm delete

The user must confirm deleting the question by selecting the *Yes* option of the confirmation dialog.

3. Question is deleted and system shows informative message

The system shows a message to inform the user of the successful action. By deleting a question, the question's possible answers are also deleted. If the question belongs to an invitation quiz, it will be deleted from that quiz too.

Delete multiple Questions at once

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *Question's* page

Flow of scenario

1. Select one or more questions through their check-boxes
2. Press *Delete Selected* Option
3. Confirm Delete
4. Selected Questions are deleted and system shows informative message

Detailed Description of flow of events

1. Select one or more questions through their check-boxes

Each panel that represents a question contains a check-box. By selecting at least one question through its check-box, the system enables the *multiple question manipulation* options

2. Press *Delete Selected* Option

One of the multiple question manipulation options is the *Delete Selected*. By clicking this button, the system will

will pop-up a confirmation window.

3. Confirm Delete

The user must confirm deleting the selected questions by clicking the *Yes* option of the confirmation dialog.

4. Selected Questions are deleted and system shows informative message

The system shows a message to inform the user of the successful action. By deleting multiple questions, every question's possible answers are also deleted. If the questions belong to an invitation quiz, they will be deleted from that quiz too.

Delete Answer

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *create/edit question* screen
3. Question's type is *Multiple Choice*

Flow of scenario

1. Select the trash-icon button of an Answer
2. Confirm delete
3. Save Question

Detailed Description of flow of events

1. Select the trash-icon button of an Answer

Every panel that represents an answer contains a delete button in the form of a trash-icon. By selecting this option the system will pop-up a confirmation window.

2. Confirm delete

The user must confirm deleting the question by selecting the *Yes* option of the confirmation dialog. At this point the answer is only deleted locally.

3. Save Question

An answer can only exist as part of a question. To delete an answer from the database, the user must save the question by selecting the *Save* option.

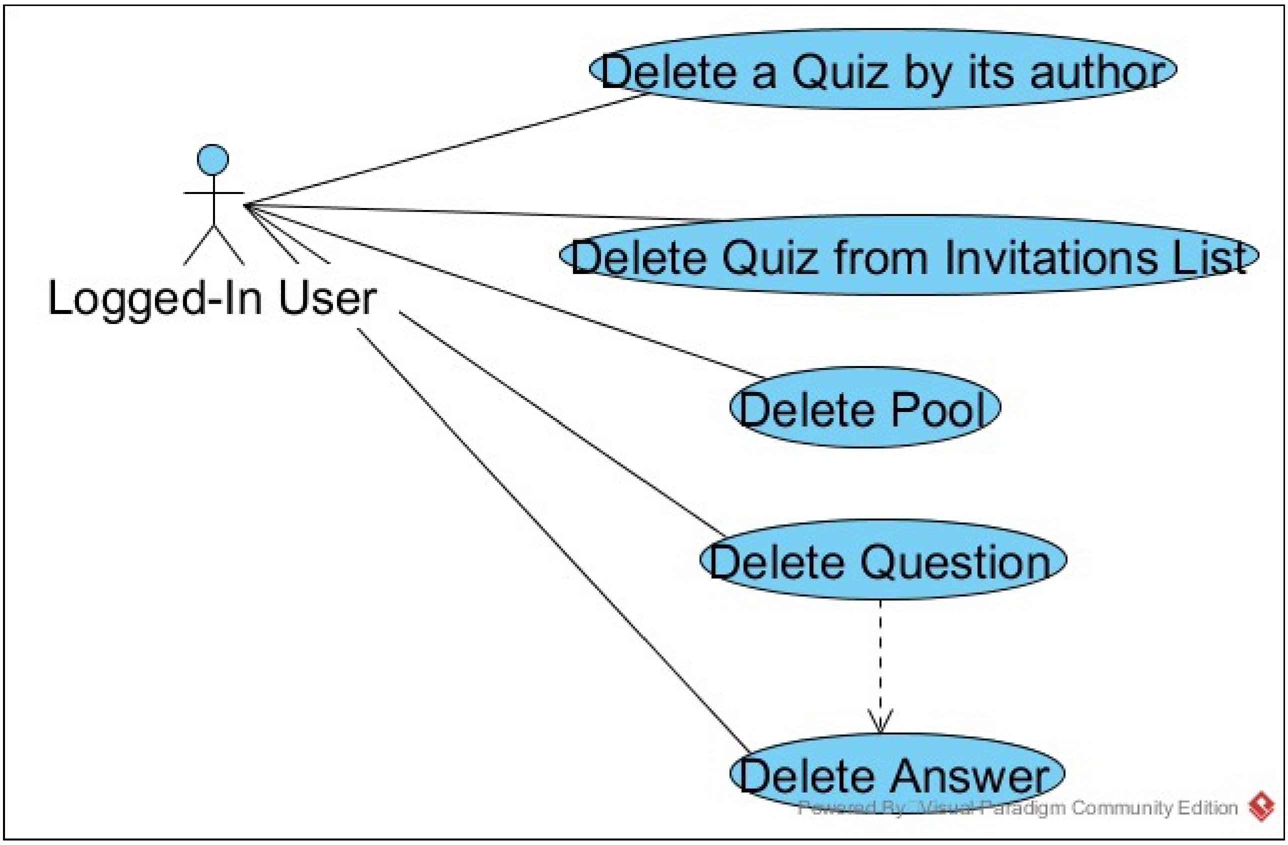

Logged-In User
Delete a Quiz by its author
Delete Quiz from Invitations List
Delete Pool
Delete Question
Delete Answer
Powered By Visual Paradigm Community Edition

Illustration 12: Graphic Representation of Scenarios of use case : Delete

3.2.2.3 Scenarios of use case : Edit

Edit Quiz Settings

Actors: Client

Preconditions:

1. Client is logged in the system
2. Quiz to edit has already been created

Basic Flow of scenario

1. Select the *Edit* quiz settings option
2. Alter data
3. Select Save option
4. Quiz is saved with new characteristics

Alternative Flow of scenario

4. a. Input is not accepted and system informs the user accordingly

Detailed Description of Basic flow of events

1. Select the *Edit* quiz settings option

In order to open the Edit quiz settings dialog, the user must select the suitable option. This option can be found

- at each panel that represents a quiz (edit icon button)
- at the *Jumbotron* area of question's page, when a quiz is selected

2. Alter data

The user can edit the title, subject, publicity and reward type of a quiz through the given form.

3. Select Save option

By selecting the *Save* option, the system validates the user's input.

4. Quiz is saved with new characteristics

If the input data is valid, the system saves the quiz with it's new characteristics in the database. The edit quiz settings dialog closes and a message appears to inform the user of the successful changes.

Detailed Description of Alternative flow of events

4. a. Input is not accepted and system informs the user accordingly

If the input data is invalid, the system shows error messages indicating what went wrong. User must enter new data into the fields that input wasn't accepted. The flow continues from step 2 (Alter data).

Edit Pool Settings

Actors: Client

Preconditions:

1. Client is logged in the system
2. Pool to edit has already been created

Basic Flow of scenario

1. Select the *Edit* pool settings option
2. Alter data
3. Select Save option
4. Pool is saved with new characteristics

Alternative Flow of scenario

4. a. Input is not accepted and system informs the user accordingly

Detailed Description of Basic flow of events

1. Select the *Edit* pool settings option

In order to open the Edit pool settings dialog, the user must select the suitable option. This option can be found

- at each panel that represents a pool (edit icon button)
- at the *Jumbotron* area of question's page, when a pool is selected

2. Alter data

The user can only edit the title of a pool through the given form.

3. Select Save option

By selecting the *Save* option, the system validates the user's input.

4. Pool is saved with new characteristics

If the input data is valid, the system saves the pool with it's new title in the database. The edit pool settings dialog closes and a message appears to inform the user of the successful changes.

Detailed Description of Alternative flow of events

4. a. Input is not accepted and system informs the user accordingly

If the new title is invalid, the system shows error messages indicating the restrictions. User must enter a new title in order to continue. The flow continues from step 2 (Alter data).

Edit Question's features

Actors: Client

Preconditions:

1. Client is logged in the system
2. Question to edit has already been created

Basic Flow of scenario

1. Select the *Edit Question* option
2. Edit question's features
3. Select *Save* option
4. Question is saved with the new features

Alternative Flow of scenario

4. a. Input is not accepted and system informs the user accordingly

Detailed Description of Basic flow of events

1. Select the *Edit Question* option

The user can open the Edit Question dialog by selecting the suitable option. This button can be found in every panel that represents a question (pencil icon)

2. Edit question's features

A question's features may differ according to the type of quiz it belongs to (if any) and the type of question that is selected. Basic features are the title, comment and type of the question.

3. Select *Save* option

By selecting the *Save* option the system validates the user's input.

4. Question is saved with the new features

If the given input is valid, the system saves the edited question in the database. The edit question dialog closes and a message appears to inform the user of the successful changes.

Detailed Description of Alternative flow of events

4. a. Input is not accepted and system informs the user accordingly

If the input is invalid, the system shows error messages indicating what went wrong. User must enter new data into the fields that input wasn't accepted. The flow continues from step 2 (Edit question's features).

Edit Answer's features

Actors: Client

Preconditions:

1. Client is logged in the system
2. Answer to edit has already been created
3. Answer belongs to a question of *Multiple Choice* type

Extends: Scenario *Edit Question's features*

Basic Flow of scenario

1. Select the *Edit* question option
2. Edit answer's features
3. Select *Save* option
4. Answer is saved through it's question

Alternative Flow of scenario

4. a. Input is not accepted and system informs the user accordingly

Detailed Description of Basic flow of events

1. Select the *Edit Question* option

The user can open the Edit Question dialog by selecting the suitable option. This button can be found in every panel that represents a question (pencil icon)

2. Edit answer's features

The user can edit the title of an answer and the points (if quiz type allows it).

3. Select *Save* option

By selecting the Save button the system validates the new data.

4. Answer is saved through it's question

If the user's input is valid, the system saves the question. The edited answer is saved though the question it belongs to.

Detailed Description of Alternative flow of events

4. a. Input is not accepted and system informs the user accordingly

If the input is invalid, the system shows error messages indicating what went wrong. User must enter new data into the fields that input wasn't accepted. The flow continues from step 2 (Edit answer's features).

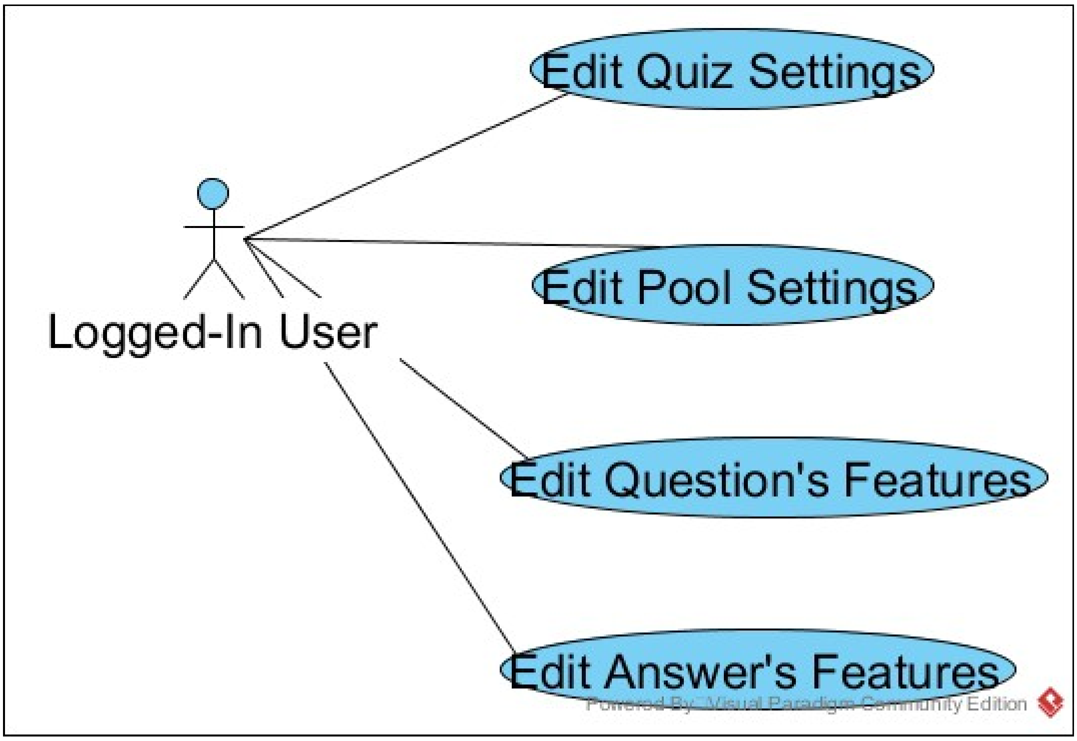


Illustration 13: Graphic Representation of Scenarios of use case : Edit

3.2.2.4 Scenarios of use case : Add

Add New Question to Quiz

Actors: Client

Preconditions:

1. Client is logged in the system
2. The quiz that a question will be added to already exists

Extends: Scenario *Create Question*

Flow of scenario

1. Select suitable option to go to the question's of quiz page
2. Scenario: *Create Question*

Detailed Description of flow of events

1. Select suitable option to go to the question's of quiz page

To add a question in a quiz, the user must be in the screen that shows only the questions that belong to that specific quiz. This can be achieved by

- clicking the menu option *Questions* → selecting a quiz from the drop down menu
- selecting the *Questions: 'amount of questions'* option of the panel that represents a quiz

2. Scenario: *Create Question*

(see scenario Create Question)

Add multiple questions to a Quiz

Actors: Client

Preconditions:

1. Client is logged in the system
2. The questions about to be added already exist
3. Client is in Question's page

Basic Flow of scenario

1. User selects at least one question
2. Press *Add Selected to Quiz* option
3. Select a quiz from the given list
4. Click *Save* option

Alternative Flow of scenario

3. a. 1. Select the +Create New Quiz option
   a. 2. Scenario : Create Quiz
   a. 3. Flow continues from step 4 (Click *Save* option)

Detailed Description of Basic flow of events

1. User selects at least one question

Every panel that represents a question contains a check-box. By clicking this check-box the question is considered 'selected'.

2. Press *Add Selected to Quiz* option

When at least one question is selected, the options for manipulating multiple elements are enabled. By selecting the *Add Selected to Quiz* option, a dialog appears. This dialog contains a list with all the available quizzes, a search input field and a create new quiz button

3. Select a quiz from the given list

The user can select a quiz from the available list by clicking on it.

4. Click *Save* option

By selecting the *Save* button the system adds the questions to the selected quiz. A message appears to inform the user of the changes and a link is provided that redirects to the question's of the selected quiz page.

Add New Question to Pool

Actors: Client

Preconditions:

1. Client is logged in the system
2. Pool already exists

Extends: Scenario *Create Question*

<u>Flow of scenario</u>

1. Select suitable option to go to the question's of pool page
2. *Scenario: Create Question*

<u>**Detailed Description of flow of events**</u>

1. <u>Select suitable option to go to the question's of pool page</u>

To add a question in a pool, the user must be in the screen that shows only the questions that belong to that specific pool. This can be achieved by

- clicking the menu option *Questions* → selecting a pool from the drop down menu
- selecting the *Questions: 'amount of questions'* option of the panel that represents a pool

2. <u>Scenario: *Create Question*</u>

(see scenario Create Question)

Add multiple questions to a Pool

Actors: Client

Preconditions:

1. Client is logged in the system
2. The questions about to be added already exist
3. Client is in Question's page

System Restrictions:

<u>Basic Flow of scenario</u>

1. User selects at least one question

2. Press *Add Selected to Pool* option
3. Select a pool from the given list
4. Click *Save* option

Alternative Flow of scenario

3. a. 1. Select the +*Create New Pool* option

   a. 2. Scenario : Create Pool

   a. 3. Flow continues from step 4 (Click *Save* option)

Detailed Description of Basic flow of events

1. User selects at least one question

Every panel that represents a question contains a check-box. By clicking this check-box the question is considered 'selected'.

2. Press *Add Selected to Pool* option

When at least one question is selected, the options for manipulating multiple elements are enabled. By selecting the *Add Selected to Pool* option, a dialog appears. This dialog contains a list with all the available pools, a search input field and a create new pool button.

3. Select a pool from the given list

The user can select a pool from the available list by clicking on it.

4. Click *Save* option

By selecting the *Save* button the system moves the questions to the selected pool. A message appears to inform the user of the changes and a link is provided that redirects to the question's of the selected pool page.

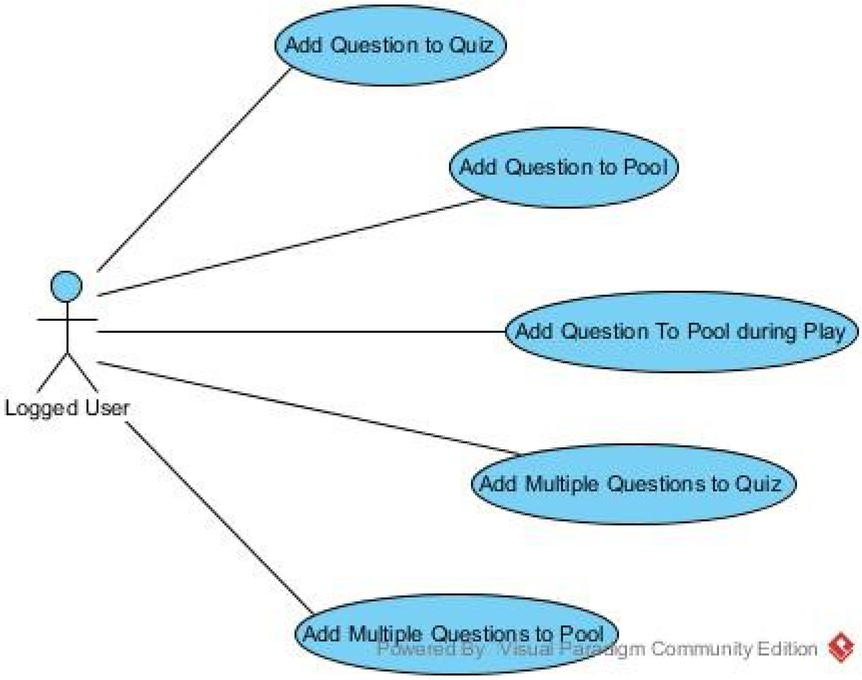

Add Question to Quiz
Add Question to Pool
Add Question To Pool during Play
Logged User
Add Multiple Questions to Quiz
Add Multiple Questions to Pool
Powered By Visual Paradigm Community Edition

Illustration 14: Graphic Representation of Scenarios of use case : Add

3.2.2.5 Scenarios of use case : Copy

Copy a question from an invitation-quiz to a pool during play-session

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the play screen
3. Play-quiz is public

Basic Flow of events

1. Select the +Add to Pool option
2. Select a pool from the given list
3. Click *Save* option

Alternative Flow of scenario

3. a. 1. Select the *+Create New Pool* option

   a. 2. Scenario : Create Pool

   a. 3. Flow continues from step 3 (Click *Save* option)

Detailed Description of Basic flow of events

1. Select the +Add to Pool option

The user must select the +Add to Pool option located in the header of each question during taking a quiz. This dialog contains a list with all the available pools, a search input field and a create new pool button.

2. Select a pool from the given list

The user can select a pool from the available list by clicking on it.

3. Click *Save* option

By selecting the *Save* button the system moves the questions to the selected pool. A message appears to inform the user of the changes and a link is provided that redirects to the question's of the selected pool page.

3.2.2.6 Scenarios of use case : Send

Send invitation for a quiz to one or more users

Actors: Client

Preconditions:

1. Client is logged in the system
2. Client is in the *My Quizzes* page

System Restrictions:

An invitation for a quiz cannot be sent twice to the same user. If an invitation has already been sent, the system disables the check-box of the receiver and shows an informative note 'Invitation sent!'

Basic Flow of events

1. Press the *Send Invitation* button of the quiz
2. Select one ore more users from the given list
3. Click the *Send* option

Detailed Description of Basic flow of events

1. Press the *Send Invitation* button of the quiz

Every panel that represents a quiz contains a Send Invitation option (envelope icon). When user selects this option, a dialog appears containing a list with all available users.

2. Select one ore more users from the given list

The user can select one or more other users from the list by clicking the check-boxes.

3. Click the *Send* option

By selecting the Send option, the system will send an invitation for that specific quiz to each of the selected users. The dialog closes and an informative message appears.

3.2.2.7 Scenarios of use case : Play

Play Quiz

Actors: Client

Preconditions:

1. Client is logged in the system

Basic Flow of events

1. Redirect to play screen
2. Select the *Start Quiz* button
3. Select/Enter an answer
4. Browse through questions
5. Submit answers

6. Close dialog

Detailed Description of Basic flow of events

1. Redirect to play screen

A user can redirect to the play screen of a quiz by one of the following cases

- click the *My Quizzes* option from the menu → select the *Play* button of a quiz
- click the *Invitations* option from the menu → select the *Play* button of a quiz

2. Select the *Start Quiz* button

The current screen shows the intro of a play session which contains some information regarding the quiz about to be played and the *Start Quiz* button. By selecting this button the game starts and the first question appears.

3. Select/Enter an answer

According to the type of the quiz and the question, the user shall select an answer from the list or enter an answer through the input filed.

4. Browse through questions

The user can browse through the questions by selecting the Next and Previous buttons.

5. Submit answers

At any point the player can submit the given answers by selecting the *Submit* option.

6. Close dialog

After submitting the answers the user can close the dialog.

3.2.2.8 Scenarios of use case : View Results

View the results of a quiz a play-session

Actors: Client

Preconditions:

1. Client is logged in the system

Basic Flow of events

1. Go to *View Results* page
2. Select a play-session from the available list
3. Browse through questions

Alternative Flow of events

3. a. Mark a *Free Text* answer as correct

Detailed Description of Basic flow of events

1. Go to *View Results* page

The user can go to View Results page by selecting the corresponding option from the menu.

2. Select a play-session from the available list

The *View Results* page provides a list with every play-session that the logged user is authorized to view. The user can select a play-session from this list in order to see the player's answers.

3. Browse through questions

The user can browse through the questions and view the player's answer's with the arrow options.

Detailed Description of Alternative Flow of events

3. a. Mark a *Free Text* answer as correct

   In case that the quiz has a reward type with points and a question is type of *Free Text,* the user that sees the results has the authority to mark the player's answer as correct by selecting the suitable check-box.

3.2.2.9 Scenarios of use case : Log In

Log to Account

Actors: Client

Preconditions:

1. Client is not logged in the system
2. Client has a registered user

Basic Flow of events

1. Visit log in screen of the system
2. Fill log in form
3. Select *Log In* option
4. System redirects to My Quizzes page

Alternative Flow of Events

4. a. System asks for valid input

Detailed Description of Basic Flow of events

1. Visit log in screen of the system

The first screen that a not logged client sees is the log in screen

2. Fill log in form

This screen provides a log in form that the user needs to fill.

3. Select *Log In* option

After filling the form, the user must select the *Log In* option in order to log to the account. When this button is pressed the system validates the user's input.

4. System redirects to My Quizzes page

If the given input is valid, the system redirects to the *My Quizzes* page and a welcoming message appears.

Detailed Description of Alternative Flow of Events

5. a. System asks for valid input

If the given input is invalid, the system shows error messages indicating what went wrong. The client must re-fill the fields where the error occurred and follow all steps from step 2 (Fill log in form)

3.2.3 Class Diagram

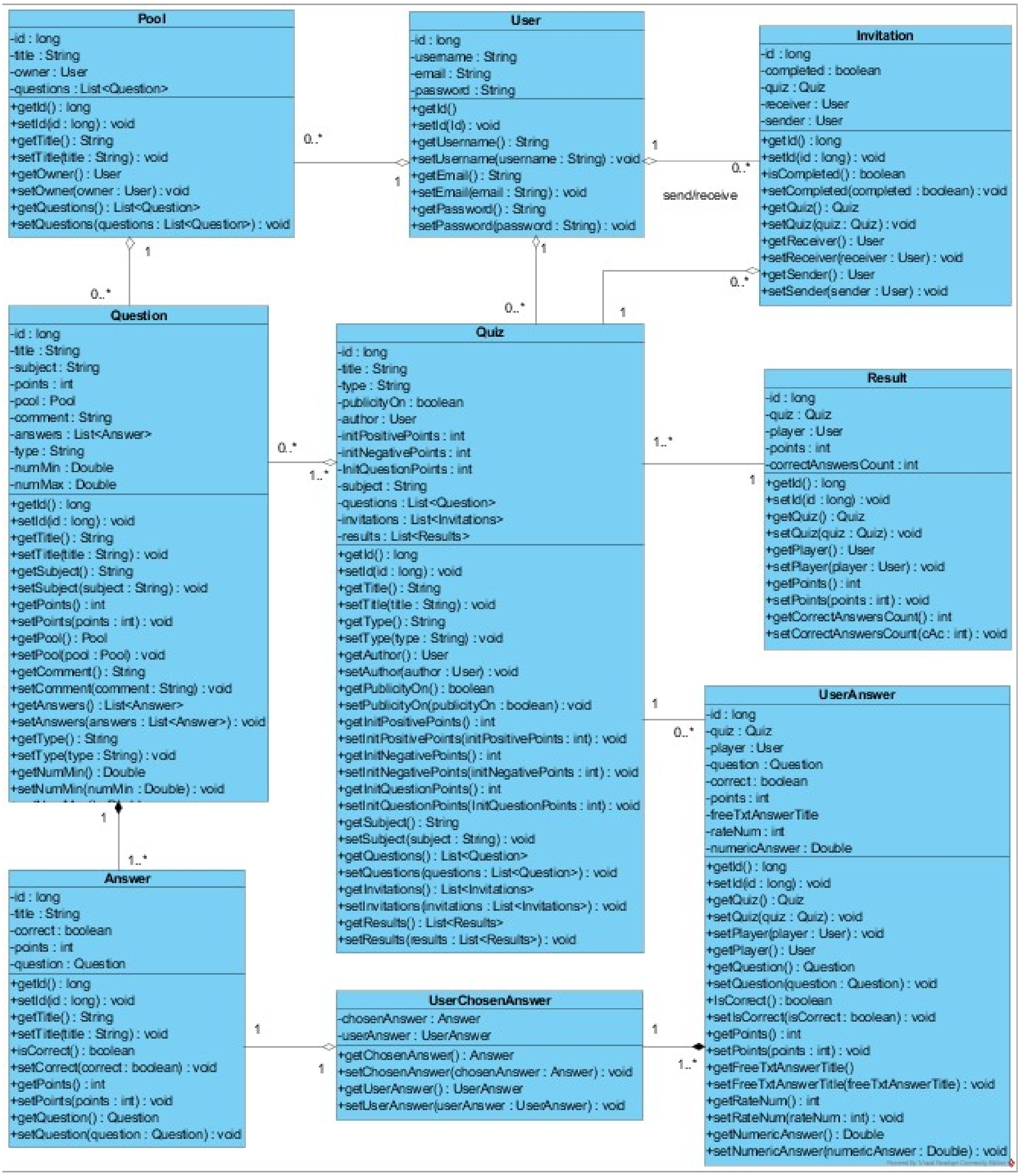

Pool
-id : long
-title : String
-owner : User
-questions : List<Question>
+getId() : long
+setId(id : long) : void
+getTitle() : String
+setTitle(title : String) : void
+getOwner() : User
+setOwner(owner : User) : void
+getQuestions() : List<Question>
+setQuestions(questions : List<Question>) : void
User
-id : long
-username : String
-email : String
-password : String
+getId()
+setId(Id) : void
+getUsername() : String
+setUsername(username : String) : void
+getEmail() : String
+setEmail(email : String) : void
+getPassword() : String
+setPassword(password : String) : void
Invitation
-id : long
-completed : boolean
-quiz : Quiz
-receiver : User
-sender : User
+getId() : long
+setId(id : long) : void
+isCompleted() : boolean
+setCompleted(completed : boolean) : void
+getQuiz() : Quiz
+setQuiz(quiz : Quiz) : void
+getReceiver() : User
+setReceiver(receiver : User) : void
+getSender() : User
+setSender(sender : User) : void
0..*
1
1
0..*
send/receive
0..*
1
0..*
1
0..*
1
Question
-id : long
-title : String
-subject : String
-points : int
-pool : Pool
-comment : String
-answers : List<Answer>
-type : String
-numMin : Double
-numMax : Double
+getId() : long
+setId(id : long) : void
+getTitle() : String
+setTitle(title : String) : void
+getSubject() : String
+setSubject(subject : String) : void
+getPoints() : int
+setPoints(points : int) : void
+getPool() : Pool
+setPool(pool : Pool) : void
+getComment() : String
+setComment(comment : String) : void
+getAnswers() : List<Answer>
+setAnswers(answers : List<Answer>) : void
+getType() : String
+setType(type : String) : void
+getNumMin() : Double
+setNumMin(numMin : Double) : void
Quiz
-id : long
-title : String
-type : String
-publicityOn : boolean
-author : User
-initPositivePoints : int
-initNegativePoints : int
-InitQuestionPoints : int
-subject : String
-questions : List<Question>
-invitations : List<Invitations>
-results : List<Results>
+getId() : long
+setId(id : long) : void
+getTitle() : String
+setTitle(title : String) : void
+getType() : String
+setType(type : String) : void
+getAuthor() : User
+setAuthor(author : User) : void
+getPublicityOn() : boolean
+setPublicityOn(publicityOn : boolean) : void
+getInitPositivePoints() : int
+setInitPositivePoints(initPositivePoints : int) : void
+getInitNegativePoints() : int
+setInitNegativePoints(initNegativePoints : int) : void
+getInitQuestionPoints() : int
+setInitQuestionPoints(InitQuestionPoints : int) : void
+getSubject() : String
+setSubject(subject : String) : void
+getQuestions() : List<Question>
+setQuestions(questions : List<Question>) : void
+getInvitations() : List<Invitations>
+setInvitations(invitations : List<Invitations>) : void
+getResults() : List<Results>
+setResults(results : List<Results>) : void
0..*
1..*
1..*
1
Result
-id : long
-quiz : Quiz
-player : User
-points : int
-correctAnswersCount : int
+getId() : long
+setId(id : long) : void
+getQuiz() : Quiz
+setQuiz(quiz : Quiz) : void
+getPlayer() : User
+setPlayer(player : User) : void
+getPoints() : int
+setPoints(points : int) : void
+getCorrectAnswersCount() : int
+setCorrectAnswersCount(cAc : int) : void
1
0..*
UserAnswer
-id : long
-quiz : Quiz
-player : User
-question : Question
-correct : boolean
-points : int
-freeTxtAnswerTitle
-rateNum : int
-numericAnswer : Double
+getId() : long
+setId(id : long) : void
+getQuiz() : Quiz
+setQuiz(quiz : Quiz) : void
+setPlayer(player : User) : void
+getPlayer() : User
+getQuestion() : Question
+setQuestion(question : Question) : void
+IsCorrect() : boolean
+setIsCorrect(isCorrect : boolean) : void
+getPoints() : int
+setPoints(points : int) : void
+getFreeTxtAnswerTitle()
+setFreeTxtAnswerTitle(freeTxtAnswerTitle) : void
+getRateNum() : int
+setRateNum(rateNum : int) : void
+getNumericAnswer() : Double
+setNumericAnswer(numericAnswer : Double) : void
1
1..*
Answer
-id : long
-title : String
-correct : boolean
-points : int
-question : Question
+getId() : long
+setId(id : long) : void
+getTitle() : String
+setTitle(title : String) : void
+isCorrect() : boolean
+setCorrect(correct : boolean) : void
+getPoints() : int
+setPoints(points : int) : void
+getQuestion() : Question
+setQuestion(question : Question) : void
1
1
UserChosenAnswer
-chosenAnswer : Answer
-userAnswer : UserAnswer
+getChosenAnswer() : Answer
+setChosenAnswer(chosenAnswer : Answer) : void
+getUserAnswer() : UserAnswer
+setUserAnswer(userAnswer : UserAnswer) : void
1
1..*

Illustration 15: Class Diagram

3.2.4 Entity Relationship Diagram

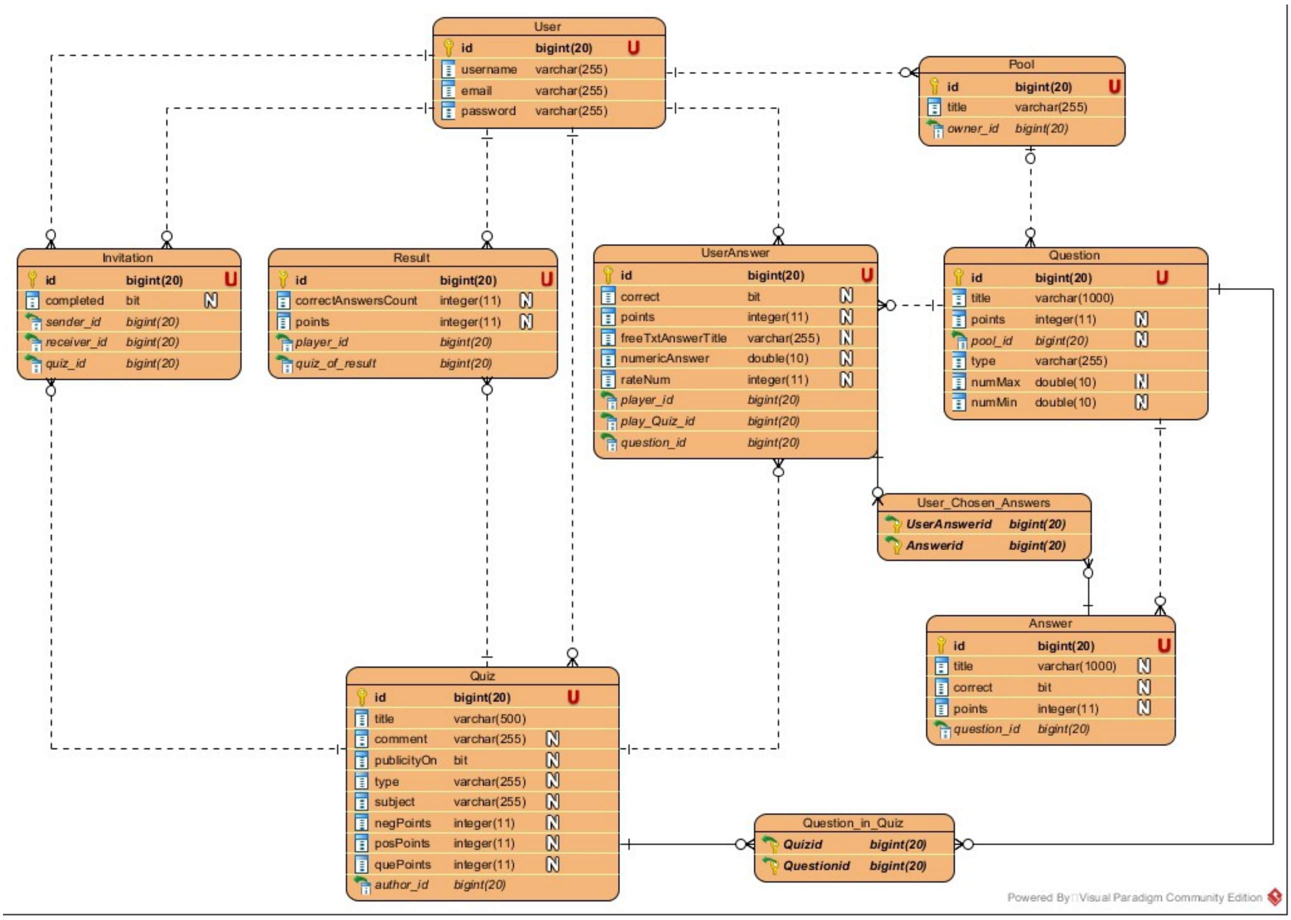
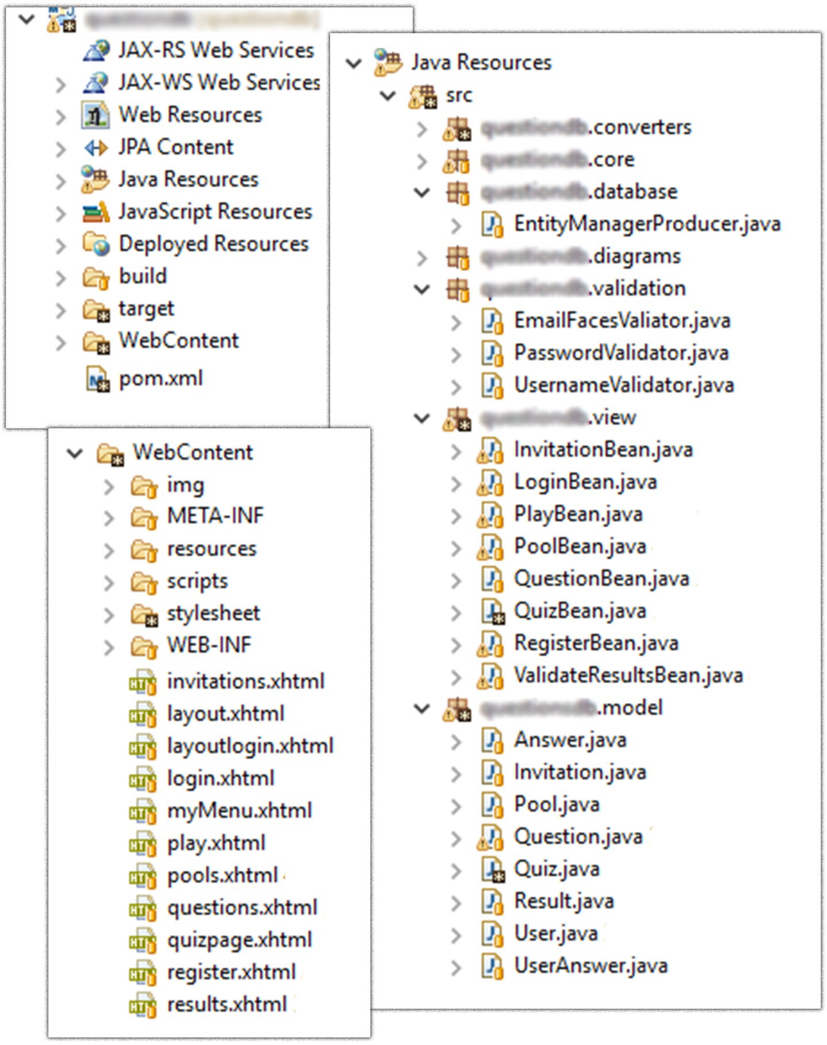


Illustration 16: Entity Relationship Diagram

3.3 Development

3.3.1 Project Structure



```
v [redacted]
    JAX-RS Web Services
  > JAX-WS Web Services
  > Web Resources
  > JPA Content
  > Java Resources
  > JavaScript Resources
  > Deployed Resources
  > build
  > target
  > WebContent
    pom.xml

v WebContent
  > img
  > META-INF
  > resources
  > scripts
  > stylesheet
  > WEB-INF
    invitations.xhtml
    layout.xhtml
    layoutlogin.xhtml
    login.xhtml
    myMenu.xhtml
    play.xhtml
    pools.xhtml
    questions.xhtml
    quizpage.xhtml
    register.xhtml
    results.xhtml

v Java Resources
  v src
    > [redacted].converters
    > [redacted].core
    v [redacted].database
      > EntityManagerProducer.java
    > [redacted].diagrams
    v [redacted].validation
      > EmailFacesValiator.java
      > PasswordValidator.java
      > UsernameValidator.java
    v [redacted].view
      > InvitationBean.java
      > LoginBean.java
      > PlayBean.java
      > PoolBean.java
      > QuestionBean.java
      > QuizBean.java
      > RegisterBean.java
      > ValidateResultsBean.java
    v [redacted].model
      > Answer.java
      > Invitation.java
      > Pool.java
      > Question.java
      > Quiz.java
      > Result.java
      > User.java
      > UserAnswer.java
```

Illustration 17: Project & Files Structure

As the picture on the left shows, the project contains six packages with each package reflecting a different set of features. The model package contains the classes that represent the system's model objects such as the User, Quiz and Question. The view package contains all java beans classes that handle the business logic of the application. The database package contains a class that handles database connectivity. Hence, in this case, the name of the package corresponds to the functionality of the classes it encloses.

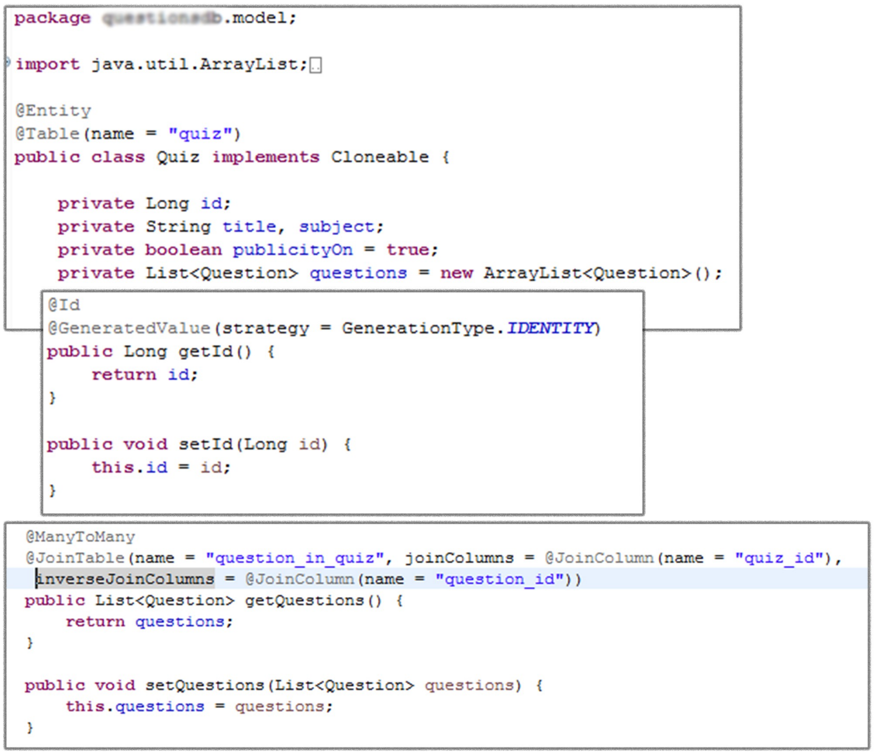
The Web Content folder consists of elements related to the pages. The *stylesheet* folder contains all css files that style the pages. The *resources* folder has reusable snippets of code that are included from pages. Images and scripts can be found under the *img* and *scripts* folders. Last but not least, the WebContent folder includes all XHTML pages.

3.3.2 Classes of Model-Objects

Every class that 'describes' a model-object of this system is written following the rules and structure of a *plain old java class* (POJO). In each class, instance variables are defined to express the state of an object (e.g the ID of an object). Every class contains the implementation of the default *getters* and *setters* to assure encapsulation. Also, in order to avoid false conclutions in comparissons between objects, these classes override the implementation of equals() and hashcode() methods.

As mentioned in chapter *2.3 Data Access Layer Technologies,* this project uses object-relational mapping technique which means that every model-object class has to be defined as an *entity*. This is achieved by using the @Entity and @Table annotations above the class declaration. To define which variable is the ID of an entity, Hibernate provides the @Id and @GeneratedValue annotations that can be placed above the Id's default getter. A number of other annotations such as @ManyToOne or @ManyToMany are also used to define joins and mapping between object relationships.

The picture that follows shows the structure of the class **Quiz** as an example of the information stated previously.

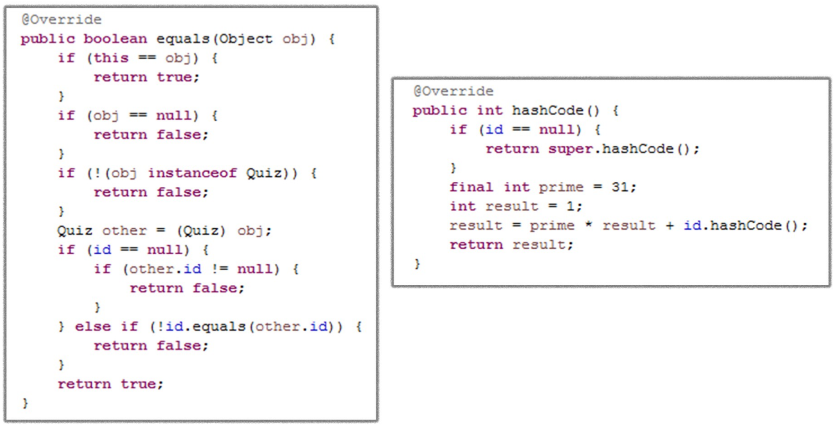


```
package [redacted].model;

import java.util.ArrayList;

@Entity
@Table(name = "quiz")
public class Quiz implements Cloneable {

    private Long id;
    private String title, subject;
    private boolean publicityOn = true;
    private List<Question> questions = new ArrayList<Question>();
```

```
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
public Long getId() {
    return id;
}

public void setId(Long id) {
    this.id = id;
}
```

```
@ManyToMany
@JoinTable(name = "question_in_quiz", joinColumns = @JoinColumn(name = "quiz_id"),
 inverseJoinColumns = @JoinColumn(name = "question_id"))
public List<Question> getQuestions() {
    return questions;
}

public void setQuestions(List<Question> questions) {
    this.questions = questions;
}
```

Illustration 18: Entity Class example



```
@Override
public boolean equals(Object obj) {
    if (this == obj) {
        return true;
    }
    if (obj == null) {
        return false;
    }
    if (!(obj instanceof Quiz)) {
        return false;
    }
    Quiz other = (Quiz) obj;
    if (id == null) {
        if (other.id != null) {
            return false;
        }
    } else if (!id.equals(other.id)) {
        return false;
    }
    return true;
}
```

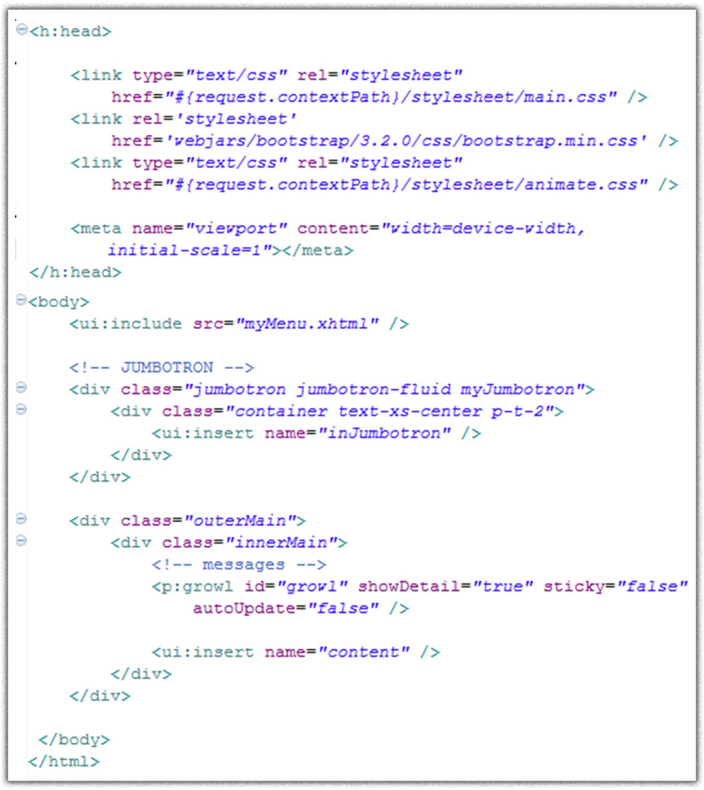
```
@Override
public int hashCode() {
    if (id == null) {
        return super.hashCode();
    }
    final int prime = 31;
    int result = 1;
    result = prime * result + id.hashCode();
    return result;
}
```

Illustration 19: Entity Class example

3.3.3 Web Pages

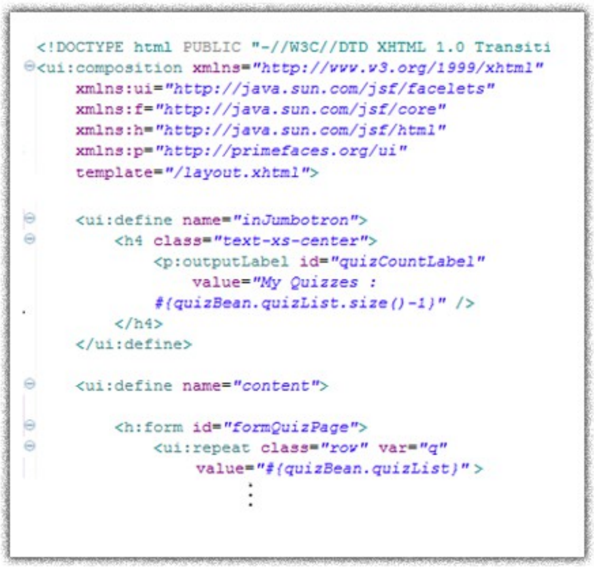
For this project, there were created two kinds of templates and each page follows one or the other according to the its needs. The first template is used in Log in and Register pages and the second one is used in all the other application pages. While template creation is not mandatory, it helps keeping the code clean and readable. A template is created once but used by many pages. It is written following the basic HTML structure and it contains all features that are common for all pages. For the areas that each page has to use differently, the template provides the ability to insert different components through the *ui:insert* tag.

From the page's point of view, using a template also comes handy. HTML's Tags like *head* and *body* are skipped as they are already defined in the template. Code writing starts with the *ui:composition* tag and it includes everything that the *body* section would have. In the areas that the template provides the *ui:insert* tag, each page can insert its code under the *ui:define* tag. The picture below shows the basic template's code and the way quiz page uses it.



```
<h:head>

    <link type="text/css" rel="stylesheet"
        href="#{request.contextPath}/stylesheet/main.css" />
    <link rel='stylesheet'
        href='webjars/bootstrap/3.2.0/css/bootstrap.min.css' />
    <link type="text/css" rel="stylesheet"
        href="#{request.contextPath}/stylesheet/animate.css" />

    <meta name="viewport" content="width=device-width,
       initial-scale=1"></meta>
</h:head>
<body>
    <ui:include src="myMenu.xhtml" />

    <!-- JUMBOTRON -->
    <div class="jumbotron jumbotron-fluid myJumbotron">
        <div class="container text-xs-center p-t-2">
            <ui:insert name="inJumbotron" />
        </div>
    </div>

    <div class="outerMain">
        <div class="innerMain">
            <!-- messages -->
            <p:growl id="growl" showDetail="true" sticky="false"
                autoUpdate="false" />

            <ui:insert name="content" />
        </div>
    </div>

 </body>
</html>
```

Illustration 20: Webpage layout



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transiti
<ui:composition xmlns="http://www.w3.org/1999/xhtml"
    xmlns:ui="http://java.sun.com/jsf/facelets"
    xmlns:f="http://java.sun.com/jsf/core"
    xmlns:h="http://java.sun.com/jsf/html"
    xmlns:p="http://primefaces.org/ui"
    template="/layout.xhtml">

    <ui:define name="inJumbotron">
        <h4 class="text-xs-center">
            <p:outputLabel id="quizCountLabel"
                value="My Quizzes :
            #{quizBean.quizList.size()-1}" />
        </h4>
    </ui:define>

    <ui:define name="content">

        <h:form id="formQuizPage">
            <ui:repeat class="row" var="q"
                value="#{quizBean.quizList}" >
                      :
```

Illustration 21: Webpage example

3.3.4 Java Beans

A javabean provides the functionality behind a web page. Every page of this application 'co-operates' with a bean class (e.g the quizBean.java handles the quizpage.xhtml, the questionBean.java handles the questionspage.xhtml and so on). When a client makes a request through a page element, this element calls the right method of it's bean. The bean generates a response and the page presents it to the client.

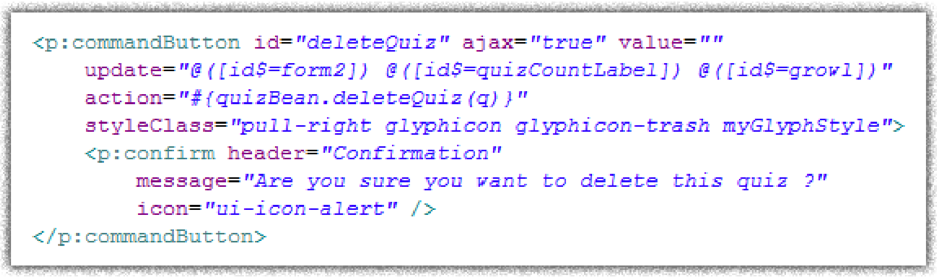
A bean class is nothing more than a regular Java class that follows certain conventions. A java bean class should:

- Be Serializable (implement java.io.Serializable interface)
- Provide the default constructor (public constructor with no arguments)
- Provide the default getters and setters (public and named according to standard naming convention)

Although beans are not supposed to interact directly with each other, in some cases that is necessary. A good example of this necessity is the need of every bean to connect to the database. In this project there is a bean (Entity manager producer) that safely opens and closes database connectivity. Every other bean that needs communication with the database must inject the Entity Manager by using the @Inject annotation.

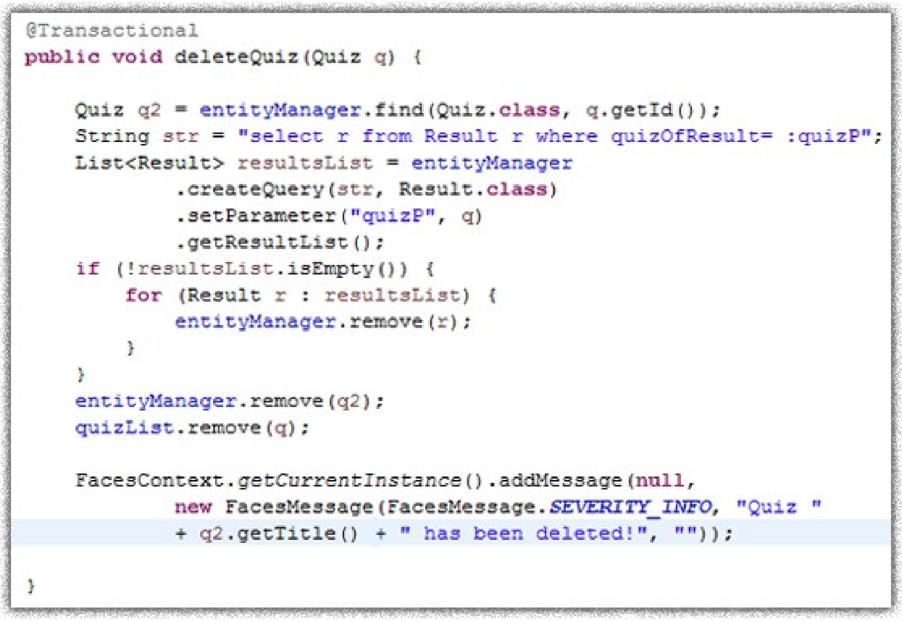
When web application uses beans that perform injection, there is the need for a bean to save a certain state for a specific amount of time. The duration specification can be defined by giving a scope to the object though the right annotation. In the example that picture *23* shows, the quizbean.java uses the @ViewScoped annotation which means that the bean will hold the state of data as long as the view is still alive. When the user switches the view (closes the tab or a window or navigates to another page), the data of the previous state will be ready for the garbage collector.

The pictures below shows how a page component can call a bean method. This example demonstrates the deletion of a quiz by its author. When the user clicks the delete button through the quizpage.xhtml (request) , the *action* attribute will be processed. In this case action is the delete method from the quizbean.java. When the method is successfully completed and the quiz is deleted from the database, a message will appear in the user's screen (response).



```
<p:commandButton id="deleteQuiz" ajax="true" value=""
    update="@([id$=form2]) @([id$=quizCountLabel]) @([id$=growl])"
    action="#{quizBean.deleteQuiz(q)}"
    styleClass="pull-right glyphicon glyphicon-trash myGlyphStyle">
    <p:confirm header="Confirmation"
        message="Are you sure you want to delete this quiz ?"
        icon="ui-icon-alert" />
</p:commandButton>
```

Illustration 22: Front-End part of delete quiz



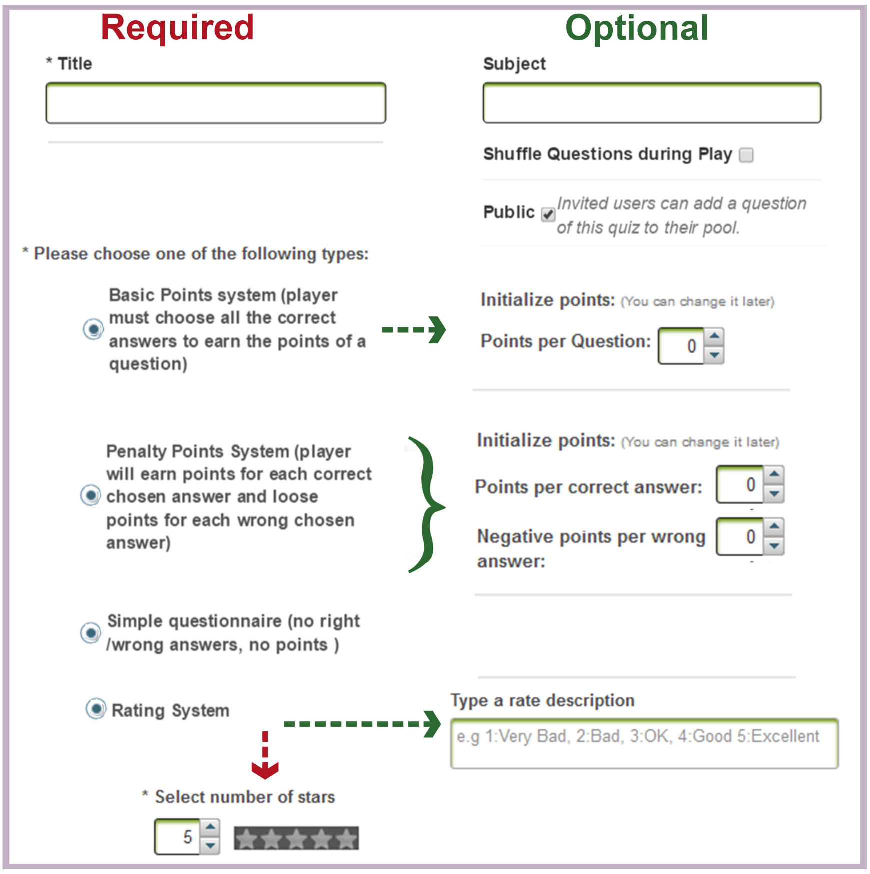
```
@Transactional
public void deleteQuiz(Quiz q) {

    Quiz q2 = entityManager.find(Quiz.class, q.getId());
    String str = "select r from Result r where quizOfResult= :quizP";
    List<Result> resultsList = entityManager
            .createQuery(str, Result.class)
            .setParameter("quizP", q)
            .getResultList();
    if (!resultsList.isEmpty()) {
        for (Result r : resultsList) {
            entityManager.remove(r);
        }
    }
    entityManager.remove(q2);
    quizList.remove(q);

    FacesContext.getCurrentInstance().addMessage(null,
            new FacesMessage(FacesMessage.SEVERITY_INFO, "Quiz "
            + q2.getTitle() + " has been deleted!", ""));

}
```

Illustration 23: Back-End part of delete quiz

3.4 System Management

3.4.2 Quiz Management

Quiz Features



Required

Optional

* Title

Subject

Shuffle Questions during Play □

Public ☑ *Invited users can add a question of this quiz to their pool.*

* Please choose one of the following types:

● Basic Points system (player must choose all the correct answers to earn the points of a question)

Initialize points: (You can change it later)

Points per Question: 0

● Penalty Points System (player will earn points for each correct chosen answer and loose points for each wrong chosen answer)

Initialize points: (You can change it later)

Points per correct answer: 0

Negative points per wrong answer: 0

● Simple questionnaire (no right /wrong answers, no points)

● Rating System

Type a rate description

e.g 1:Very Bad, 2:Bad, 3:OK, 4:Good 5:Excellent

* Select number of stars

5 ★★★★★

Illustration 24: Quiz Features

For quiz creation or editing, the system requests some mandatory and some optional information. In any kind of quiz, the **title** is an input filed that cannot remain empty and must be unique among the author's quiz list. In the case of invalid input title, the system will show an error message under the title's field, right after the *Save* button is triggered. The author can also give some additional information such as a Subject and can choose some of the optional characteristics like the Publicity and the Shuffle questions options. Unlike the required fields, the optional are not marked with the asterisk sign (*), thus they can remain empty.

The second mandatory field is the **type** of quiz which comes in the form of radio buttons. The user can choose among four options. Each type offers some additional input options that appear only when the matching radio button is selected.

The first option, as the picture on the left shows, is the **Basic Points System**. In this system, points are assigned to each question and the player can earn them by getting the answer right. The quiz-creator is able to initialise the points for each question that will be added to the current quiz. This option, though, is not required and the user can change the number of points while creating or editing each question individually.

The second type , **Penalty Points System**, allows the creator to assign points to each *answer*. The player will earn points for each correct answer but loose points for each wrong choice. Un-attempted questions will not affect the score. The system allows the creator to initialise the points and the penalty for each answer through two input spinners. The reward points spinner allows only positive numbers and the penalty spinner allows only negative numbers. Since These options are non-mandatory, the fields allow the zero option as well.

The third option is the **Simple Questionnaire**. This type of quiz doesn't have a reward type. Questions that belong to a simple questionnaire can't have points and their answers cannot be marked as “right” or “wrong”.

The final option is a **Rating System** quiz. This quiz is suitable for rating/ranking kind of questionnaires where the player is asked to fill some stars. In this case, the creator must give the amount of stars that will be given in every question of this quiz (required). The additional information of a rating quiz also includes the initialisation of each question comment (optional).

By selecting the Save button, the system will process the input data. If the given information is valid the quiz will be created.

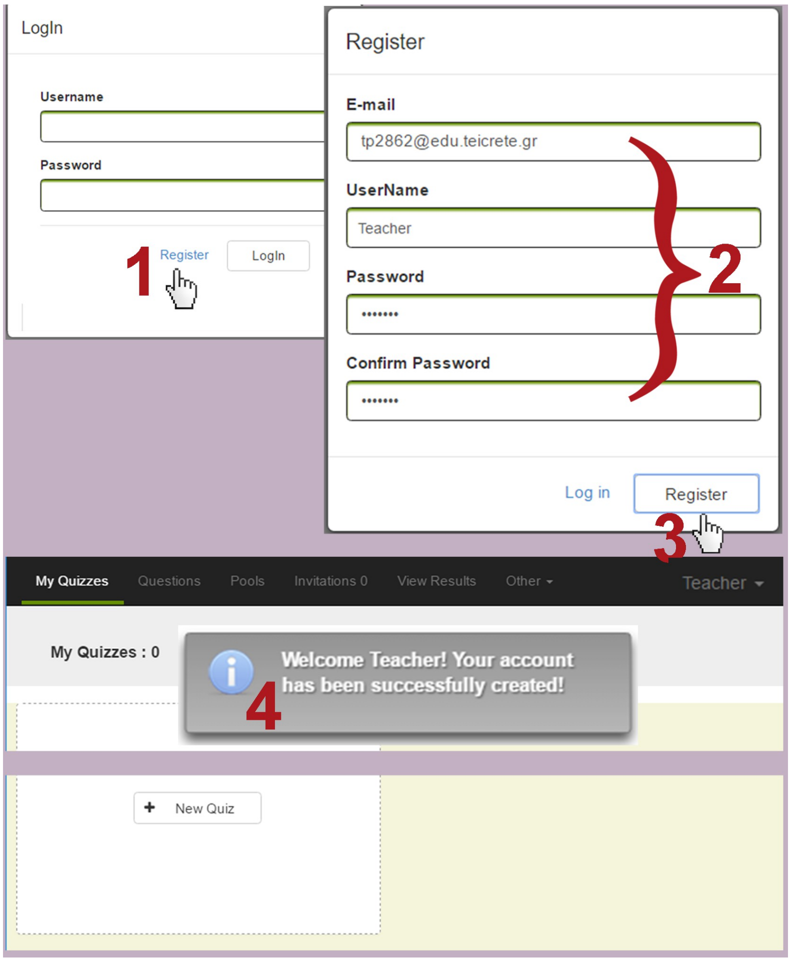
3.4.2 Case Study Freshman Questionnaire

3.4.2.1 General Scenario of Use

In this chapter, a complete scenario with real-time data will be created and demonstrated with pictures and descriptions. The chosen scenario assumes that a professor of the Department of Informatics Engineering, TEI Crete wants to use system MyQuizDB to for the Freshman Questionnaire.

The scenario will start from user **Registration**. After that, a quiz of type Simple Questionnaire and title **Freshman Questionnaire will be created**. The original university's questionnaire consists of twenty questions. For the purpose of the following examples only **4 questions will be added to the questionnaire**. Each question will be of different type, in order to demonstrate the four kinds of questions that the MyQuizDB system offers. When the questionnaire is complete, the user (teacher) will **send it as an invitation to another user**, a student. The example continues from the point of view of the student, showing how the **questionnaire is answered**. Finally, the teacher views the answers of the student.

Registration



E-mail: tp2862@edu.teicrete.gr

UserName: Teacher

Password: •••••••

Confirm Password: •••••••

My Quizzes : 0

Welcome Teacher! Your account has been successfully created!

Illustration 25: Registration

The picture on the left is a graphical representation of the scenario Create Account (see page x). This scenario describes the steps that a user has to follow in order to register.

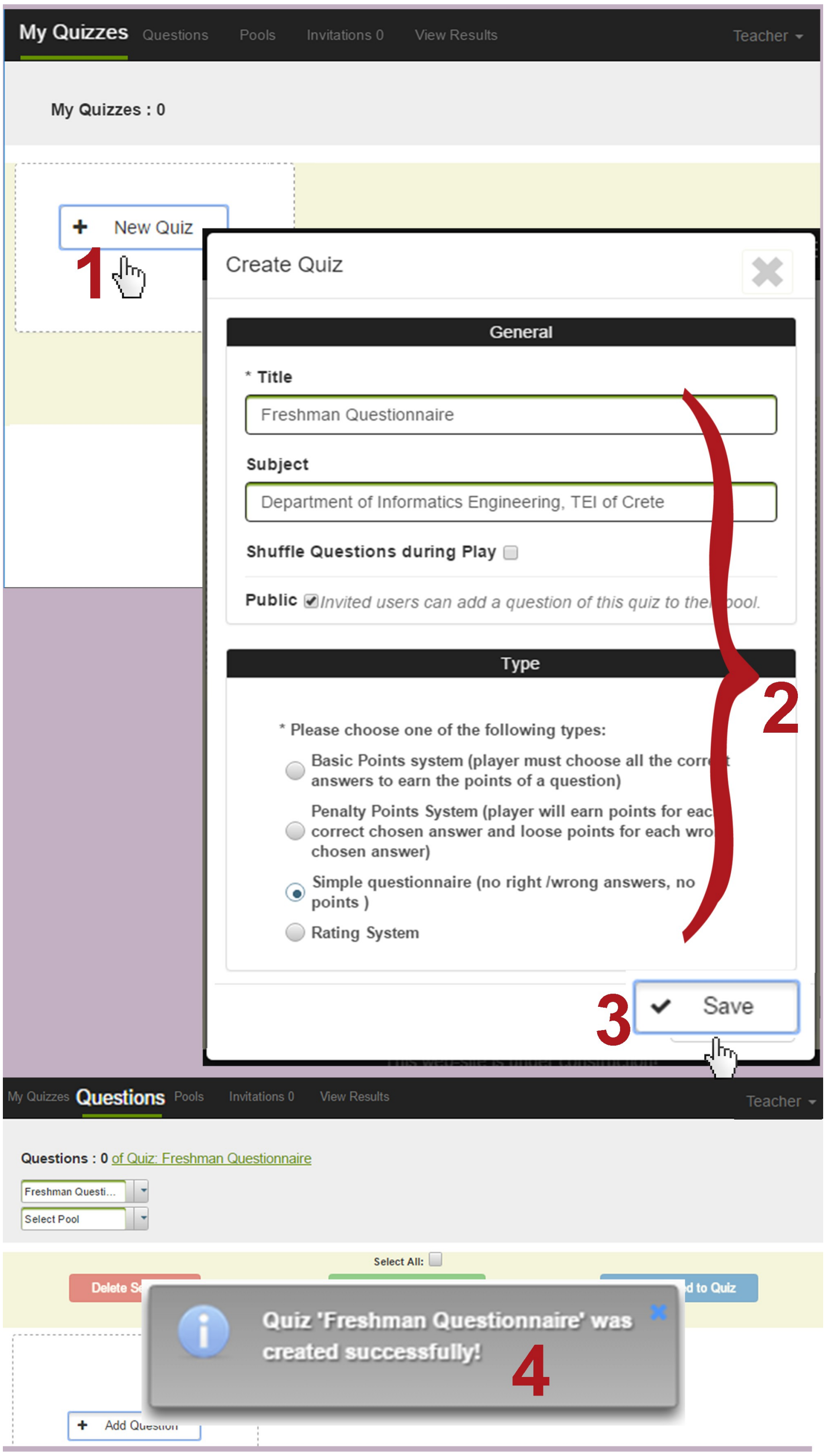
When a non-logged user visits the application's page, the first screen that is shown is the log in menu. We assume that the client of this example does not have an account so instead of logging in, the user selects the *Register* link (step 1).

Triggering the above link makes the registration environment appear. It contains a form with four required input fields. The user has to give a valid *e-mail* address, a *user-name* that is unique to the system, a *password* and the password's *confirmation* (step 2).

After filling the form, the user can submit it so that the system can evaluate the input data. To do so, the user selects the
Register button (step 3).

If the information given is not valid, the system will show error messages under the fields that were considered problematic and will not allow the account creation until the invalid input is corrected. In this example, we assume that the user gives valid data on the first try. When the data is accepted, the system redirects to to My Quizzes page and shows a welcoming message (step 4).

Create Quiz of type Simple Questionnaire



My Quizzes : 0

Create Quiz

General

* Title

Freshman Questionnaire

Subject

Department of Informatics Engineering, TEI of Crete

Shuffle Questions during Play □

Public ☑ Invited users can add a question of this quiz to their pool.

Type

* Please choose one of the following types:

○ Basic Points system (player must choose all the correct answers to earn the points of a question)

○ Penalty Points System (player will earn points for each correct chosen answer and loose points for each wrong chosen answer)

● Simple questionnaire (no right /wrong answers, no points)

○ Rating System

✓ Save

Questions : 0 of Quiz: Freshman Questionnaire

Freshman Questi...

Select Pool

Select All: □

Delete Se... | ...d to Quiz

Quiz 'Freshman Questionnaire' was created successfully!

+ Add Question

Illustration 26: Create Quiz of type Simple Questionnaire

The picture on the left is a screenshot representation of the scenario *Create Quiz* and picks up where the previous example left off.

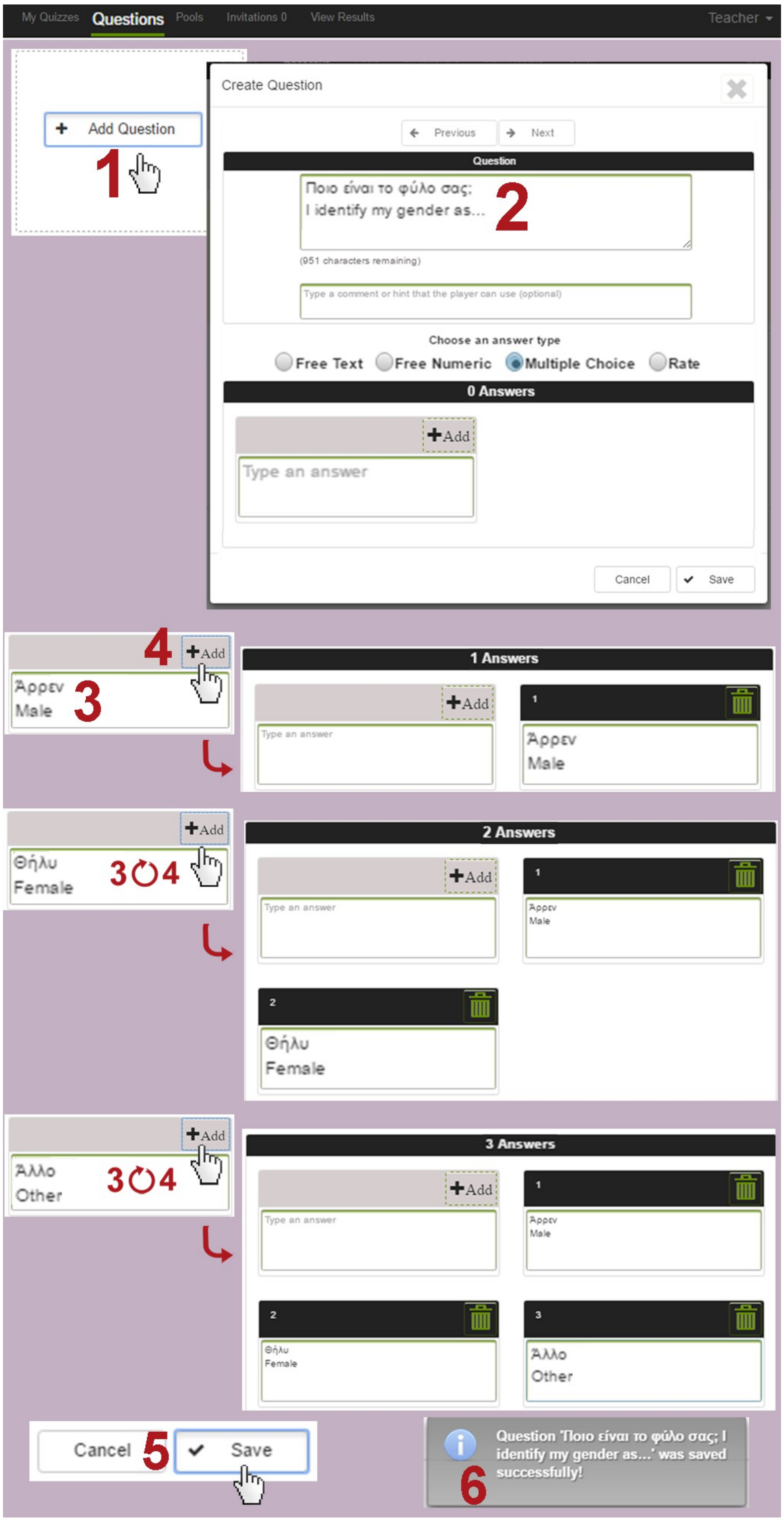
The current screen is the *My Quizzes* page. Since the user just registered, there are no quizzes yet. The page only contains a + *New Quiz* option that the user can select to create the first quiz (step 1).

When the above mentioned button is triggered, the quiz creation environment appears in a pop-up window. The user enters Freshman Questionnaire in the title input filed and the name of the institute in the subject area. The order of questions matters in this case so the option for shuffling questions will remain unchecked. The publicity option is checked by default and for this example it will remain public. Because the questionnaire is made for statistical reasons, there is no need for a points system, plus there are no right or wrong answers. Thus, the user chooses the third option Simple Questionnaire as the quiz' type (step 2).

After filling the form, the user can submit it by selecting the Save option (step 3).

By submitting the form, the system evaluates the given data. The input information in this example is valid so the system redirects to the questions page of the freshly created quiz and shows an informative message of the successful quiz creation (step 4).

Question Creation of type Multiple Choice



Ποιο είναι το φύλο σας;
I identify my gender as...

1 Άρρεν Male

2 Θήλυ Female

3 Άλλο Other

Question 'Ποιο είναι το φύλο σας; I identify my gender as...' was saved successfully!

Illustration 27: Question Creation of type Multiple Choice

Currently, the user is in the Questions (of quiz Freshman Questionnaire) environment. No questions have been created yet so the main section of the page contains only a panel with the "Add Question" button. To create the first question of this quiz the user can select his option (step 1).

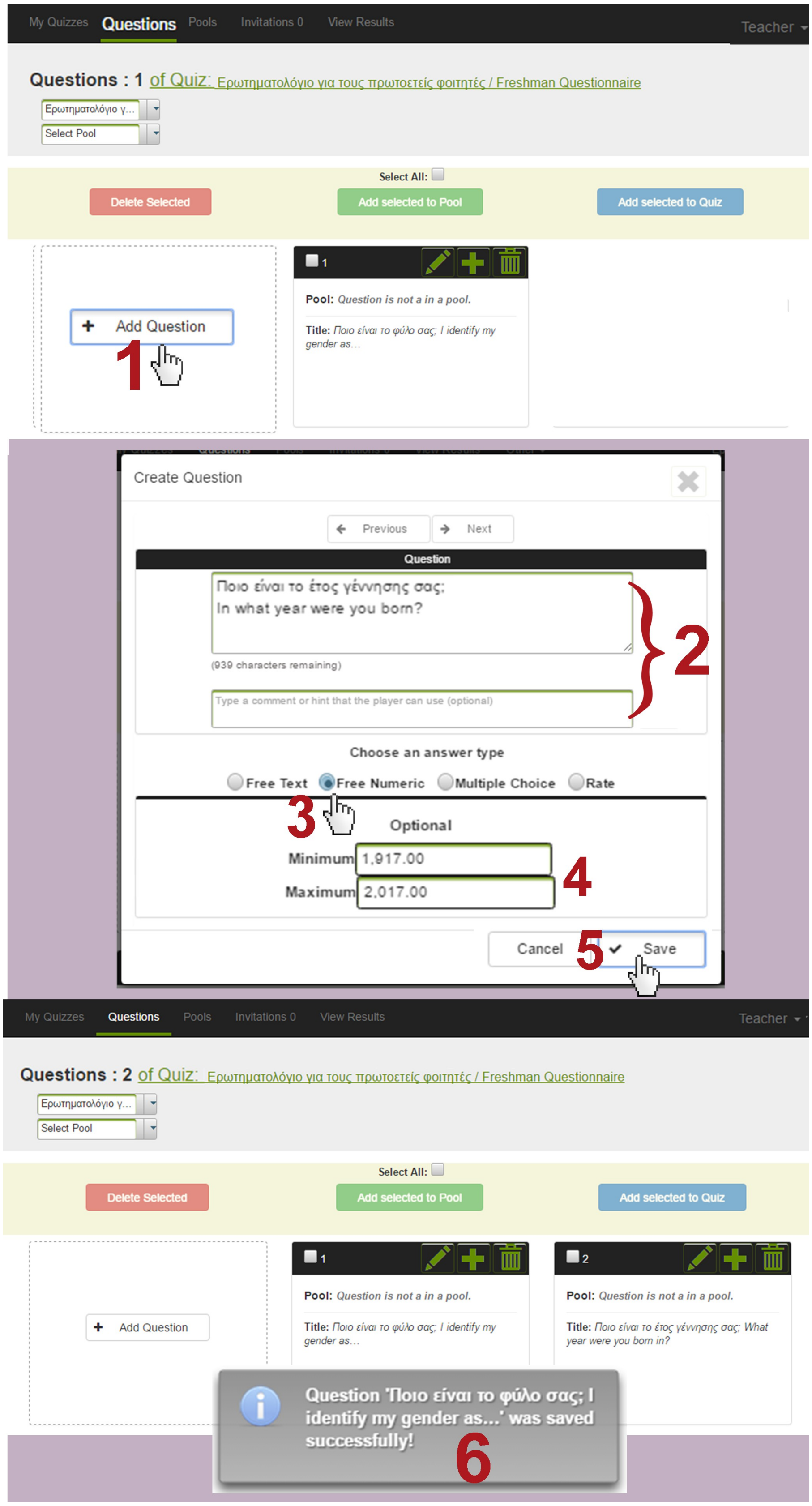
When the above mentioned button is triggered, the system pops up the question creation environment. This includes question-related options like the title (mandatory) and comment (optional) input fields. The first question of this questionnaire is about gender identity so in the title field the user enters "I identify my gender as.." (step 2).

The second section of this environment contains answer-related options. The user can choose from the given list the answer's type (*Free text, Free numeric, Multiple Choice* and *Rate*). As we can see in the picture, the option *Multiple Choice* is selected by default, so in this particular situation the user won't have to change it.

At this point, the user will add the possible answers that the player can choose from. To add the first answer the user can type the option "*Male"* in the filed with the place-holder that reads "*Type an answer*" (step 3) and select the button *Add* (step4). By selecting this option, the answer is added to the question's answer list and it appears in the form of a panel in the answer's section. Also, the number that indicates the amount of answers increases and the new answer's input field returns to its original, empty state. Steps 3 and 4 can be repeated for every answer that the user wants to give. For this questions, two more answers were added the *Female* and *Other.*

After adding all possible answers, the user can submit the form to save the question by selecting the *Save* option (step 5). In the example illustrated here, all given information is valid so the system closes the pop-up window and returns to the questions page. A message informs the user of the success (step 6).

Create Question of type Free Numeric



Questions : 1 of Quiz: Ερωτηματολόγιο για τους πρωτοετείς φοιτητές / Freshman Questionnaire

Ερωτηματολόγιο γ...

Select Pool

Pool: Question is not a in a pool.

Title: Ποιο είναι το φύλο σας; I identify my gender as...

Create Question

Question

Ποιο είναι το έτος γέννησης σας;
In what year were you born?

(939 characters remaining)

Type a comment or hint that the player can use (optional)

Choose an answer type

○ Free Text ● Free Numeric ○ Multiple Choice ○ Rate

Optional

Minimum 1,917.00

Maximum 2,017.00

Questions : 2 of Quiz: Ερωτηματολόγιο για τους πρωτοετείς φοιτητές / Freshman Questionnaire

Pool: Question is not a in a pool.

Title: Ποιο είναι το φύλο σας; I identify my gender as...

Pool: Question is not a in a pool.

Title: Ποιο είναι το έτος γέννησης σας; What year were you born in?

Question 'Ποιο είναι το φύλο σας; I identify my gender as...' was saved successfully!

Illustration 28: Create Question of type Free Numeric

As we can see in the picture, so far one question has been added to *Freshman Questionnaire*. To create the second question the user can press the *Add Question* option (step 1).

Just like the previous example, the user shall enter the question's title in the suitable input field. In this example the question is age-related so the title is "*In what year were you born?*" (step 2).

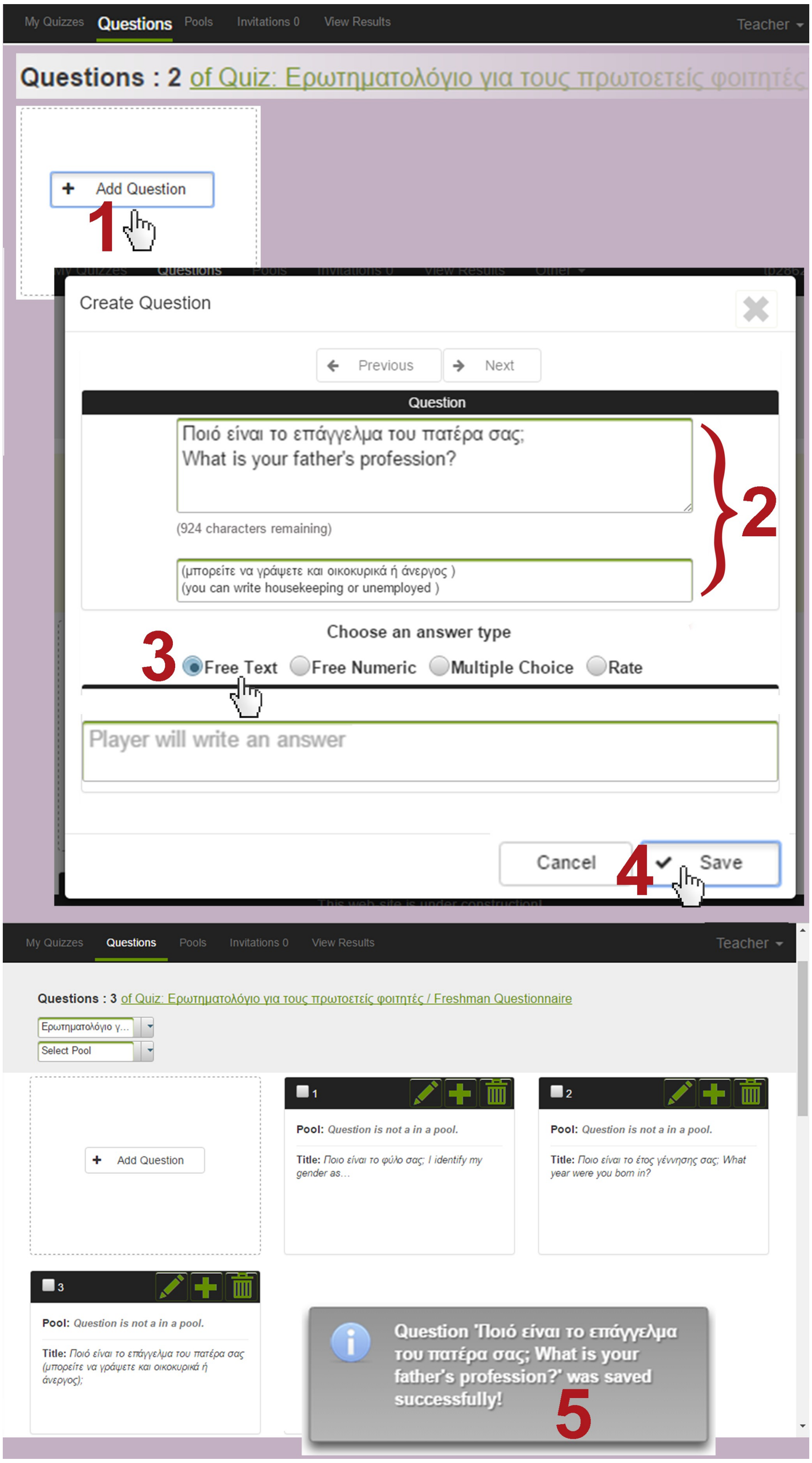
The creator of the quiz wants to limit the player's answer by accepting only numbers. To do that, the user can select the *Free Numeric* option of the answer's type list (step 3).

To restrict the player even more, the user can add a minimum and maximum limit. This options are not required but in this example the user set the acceptable input range between 1917 and 2017 (step 4).

To submit the new question form, *Save* button must be selected (step 5).

All given information was valid so by selecting the Save button, the question is created. System closes the pop-up window and the Questions page is visible again. Now the Freshman Questionnaire contains two questions. The system also informs the user of the successful question creation by showing a message (step 6).

Create Question of type Free Text



Questions : 2 of Quiz: Ερωτηματολόγιο για τους πρωτοετείς φοιτητές

Ποιό είναι το επάγγελμα του πατέρα σας;
What is your father's profession?

(924 characters remaining)

(μπορείτε να γράφετε και οικοκυρικά ή άνεργος)
(you can write housekeeping or unemployed)

Player will write an answer

Questions : 3 of Quiz: Ερωτηματολόγιο για τους πρωτοετείς φοιτητές / Freshman Questionnaire

Ερωτηματολόγιο γ...

Select Pool

1 Pool: Question is not a in a pool. Title: Ποιο είναι το φύλο σας; I identify my gender as...

2 Pool: Question is not a in a pool. Title: Ποιο είναι το έτος γέννησης σας; What year were you born in?

3 Pool: Question is not a in a pool. Title: Ποιό είναι το επάγγελμα του πατέρα σας (μπορείτε να γράφετε και οικοκυρικά ή άνεργος);

Question 'Ποιό είναι το επάγγελμα του πατέρα σας; What is your father's profession?' was saved successfully!

Illustration 29: Create Question of type Free Text

The third question of the Freshman Questionnaire is about the player's father occupation. Like all previous examples the creator must select the *Add Question* button to open the question creation environment (step 1).

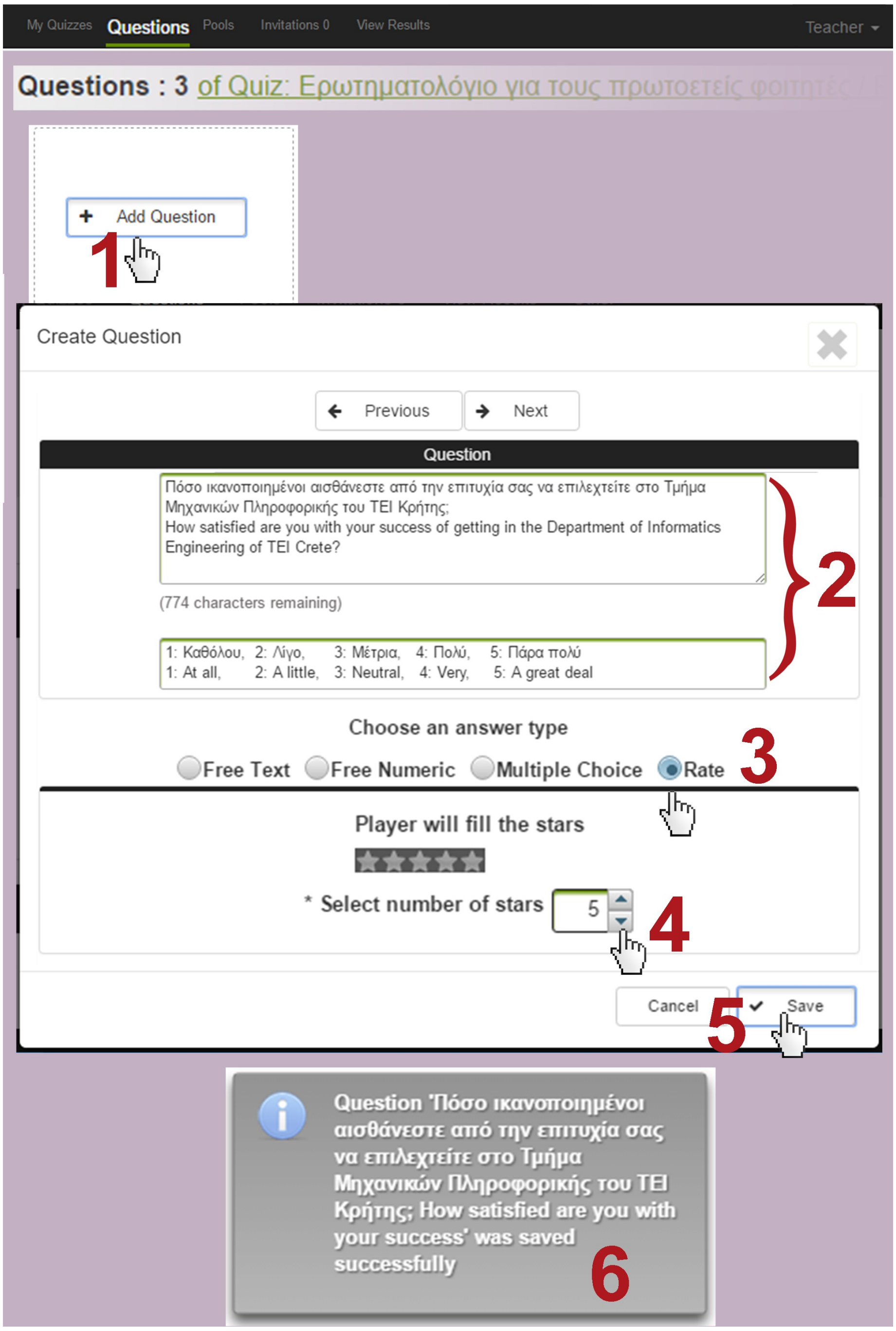
As usual, the title "What is your father's profession?" will be entered in the filed with the place-holder "*Type Question's Tile*". In the previous examples, the question's comment field (has a place-holder that reads "*Type a comment or hint that the player can use*") was not used. For this question, the quiz creator entered the comment *(you can write housekeeping or unemployed)* (step 2).

This question could be of multiple choice type but it would take too much time from the creator of the quiz to think and enter all possible jobs as possible answers. Instead, the user can set the answer's type as *Free Text* (step 3).

When the user completes step 3, the system shows a read-only input field with the place-holder "*Player will write an answer*". The user is ready to submit the question by selecting the *Save* button (step 4).

The new question has been created successfully and the system closes the question-creation environment and shows a message. The page has been refreshed and currently shows all three questions that the *Freshman Questionnaire* quiz contains (step 5).

Create Question of type Rate



Questions : 3 of Quiz: Ερωτηματολόγιο για τους πρωτοετείς φοιτητές

Πόσο ικανοποιημένοι αισθάνεστε από την επιτυχία σας να επιλεχτείτε στο Τμήμα Μηχανικών Πληροφορικής του ΤΕΙ Κρήτης;
How satisfied are you with your success of getting in the Department of Informatics Engineering of TEI Crete?

(774 characters remaining)

1: Καθόλου, 2: Λίγο, 3: Μέτρια, 4: Πολύ, 5: Πάρα πολύ
1: At all, 2: A little, 3: Neutral, 4: Very, 5: A great deal

○ Free Text ○ Free Numeric ○ Multiple Choice ● Rate

* Select number of stars 5

Question 'Πόσο ικανοποιημένοι αισθάνεστε από την επιτυχία σας να επιλεχτείτε στο Τμήμα Μηχανικών Πληροφορικής του ΤΕΙ Κρήτης; How satisfied are you with your success' was saved successfully

Illustration 30: Create Question of type Rate

For the last question example, the rate type will be used. Just like all the above cases, the user can open the pop-up question creation environment by selecting the *Add Question* option (step 1).

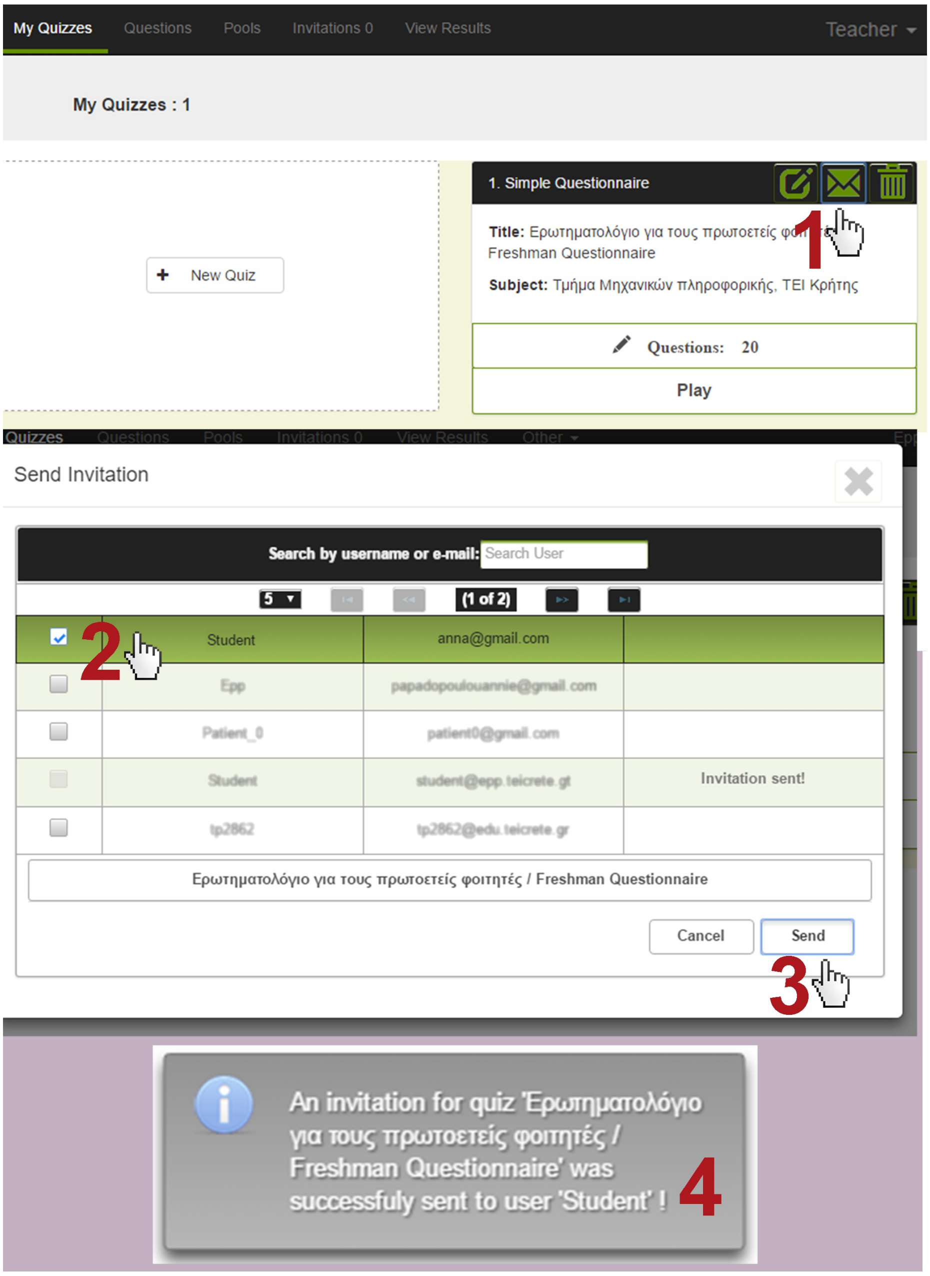
In the title input field, the user enters "*How satisfied are you with (…) ?*". In the comment field, the user can type a description of what each answer can mean. In this example, the comment explains that the range will be between 1 and 5, with 5 being the excellent point (step 2).

Since the player will be asked to rate during the quiz-taking, the creator selects the Rate option as the question's type (step 3).

When the Rate option is selected, the system shows an input-spinner, where the user can enter the amount of stars the player will be given to fill. To match the comment given above, the user enters the number 5 in the input filed (step 4).

By selecting the *Save* button the question is created (step 5). Similarly to all previous examples, the question creation environment closes and the system shows a message of the success (step 6).

Send Invitation for Quiz



Title: Ερωτηματολόγιο για τους πρωτοετείς φοι
Freshman Questionnaire

Subject: Τμήμα Μηχανικών πληροφορικής, ΤΕΙ Κρήτης

Questions: 20

Play

Send Invitation

Search by username or e-mail: Search User

5 ▾ (1 of 2)

☑	Student	anna@gmail.com	
□	Epp	papadopoulouannie@gmail.com	
□	Patient_0	patient0@gmail.com	
□	Student	student@epp.teicrete.gt	Invitation sent!
□	tp2862	tp2862@edu.teicrete.gr	

Ερωτηματολόγιο για τους πρωτοετείς φοιτητές / Freshman Questionnaire

Cancel Send

An invitation for quiz 'Ερωτηματολόγιο για τους πρωτοετείς φοιτητές / Freshman Questionnaire' was successfuly sent to user 'Student' !

Illustration 31: Send Invitation for Quiz

So far, a user with the user-name *Teacher* has registered and created a quiz of type Simple Questionnaire with title Freshman Questionnaire Twenty questions have been added to this quiz (four of them are displayed above). Now the Teacher can send this questionnaire to another user of the system. We assume that a second user has already created an account with the user-name *Student*.

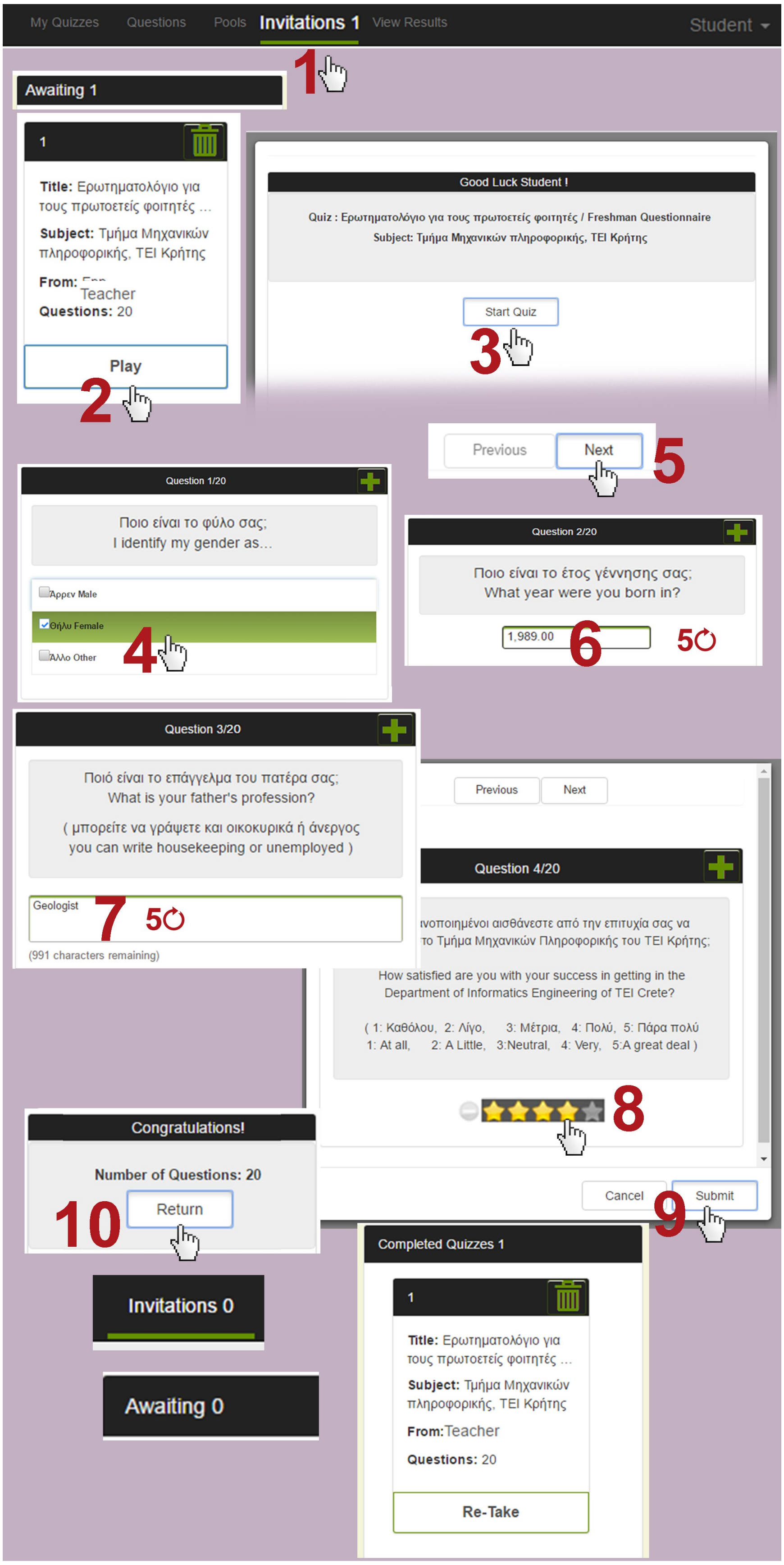
The Teacher can select the send-invitation option in the form of an envelope icon, located in the header of the panel that represents the quiz (step 1).

When this button is clicked, a pop-up window appears containing a list with all the users of the system. The Teacher can search or browse the list to find the suitable user. In this example the Student user is the first row of the list, so the Teacher selects it to continue (step 2).

After selecting the suitable receiver(s), the user can press the *Send* option, to send the invitation (step 3).

By selecting the Send option the pop-up environment closes and a message appears to inform the user of the success (step 4).

Invitation Receiver Completes Questionnaire



Awaiting 1

1

Title: Ερωτηματολόγιο για τους πρωτοετείς φοιτητές ...

Subject: Τμήμα Μηχανικών πληροφορικής, ΤΕΙ Κρήτης

From: Teacher

Questions: 20

Play

Good Luck Student !

Quiz : Ερωτηματολόγιο για τους πρωτοετείς φοιτητές / Freshman Questionnaire

Subject: Τμήμα Μηχανικών πληροφορικής, ΤΕΙ Κρήτης

Start Quiz

Previous Next

Question 1/20

Ποιο είναι το φύλο σας;
I identify my gender as...

☐ Άρρεν Male
☑ Θήλυ Female
☐ Άλλο Other

Question 2/20

Ποιο είναι το έτος γέννησης σας;
What year were you born in?

1,989.00

Question 3/20

Ποιό είναι το επάγγελμα του πατέρα σας;
What is your father's profession?

(μπορείτε να γράψετε και οικοκυρικά ή άνεργος
you can write housekeeping or unemployed)

Geologist

(991 characters remaining)

Previous Next

Question 4/20

...ινοποιημένοι αισθάνεστε από την επιτυχία σας να ...το Τμήμα Μηχανικών Πληροφορικής του ΤΕΙ Κρήτης;

How satisfied are you with your success in getting in the Department of Informatics Engineering of TEI Crete?

(1: Καθόλου, 2: Λίγο, 3: Μέτρια, 4: Πολύ, 5: Πάρα πολύ
1: At all, 2: A Little, 3:Neutral, 4: Very, 5:A great deal)

★★★★☆

Cancel Submit

Congratulations!

Number of Questions: 20

Return

Invitations 0

Awaiting 0

Completed Quizzes 1

1

Title: Ερωτηματολόγιο για τους πρωτοετείς φοιτητές ...

Subject: Τμήμα Μηχανικών πληροφορικής, ΤΕΙ Κρήτης

From: Teacher

Questions: 20

Re-Take

Illustration 32: Invitation Receiver Completes Questionnaire

The scenario represented on the left is from the point of view of the user *Student*.

When a logged user receives an invitation for a quiz, the number next to the *Invitations* menu increases by one. In this case, the user Student has one invitation from the user Teacher. To handle it, the user must select the *Invitations* option from the main menu (step 1).

The *Invitations page* is divided into two sections. The first contains the quizzes to be completed (*Awaiting*) and the second, the quizzes that have already been taken (*Completed*). Under the Awaiting section there is the invitation quiz sent by the teacher. The student must select the *Play* option of this quiz (step 2).

By selecting the above mentioned button, a pop-up window appears containing the *Game* environment. The first screen of the game is an intro screen that informs the player of the title and subject of the quiz. User can press *Start Quiz* to play (step 3).

During a play-session, the player answers one question at a time. Since the teacher *didn't* select the *Shuffle Questions* option during Freshman Questionnaire's creation, all questions created in the previous examples will appear in the same order. Hence, the first question is the gender identification. It is of type multiple choice, so the Student can answer by selecting one of the given options (step 4).

To go to the next question, the user can select *Next* (step 5).

The second question asks for birth year of the player. It is of type *Free Numeric* so the player is given an input field that accepts only numbers between 1917 and 2017. The player enters an acceptable number in the field (step 6).

To go to the next question, *step 5* shall be repeated (select *Next* option). The third question is of type *Free Text*. In this case, the player is given an input filed to type an answer. The only limitation here is that the number of characters that must not exceed the 1000 (step 7).

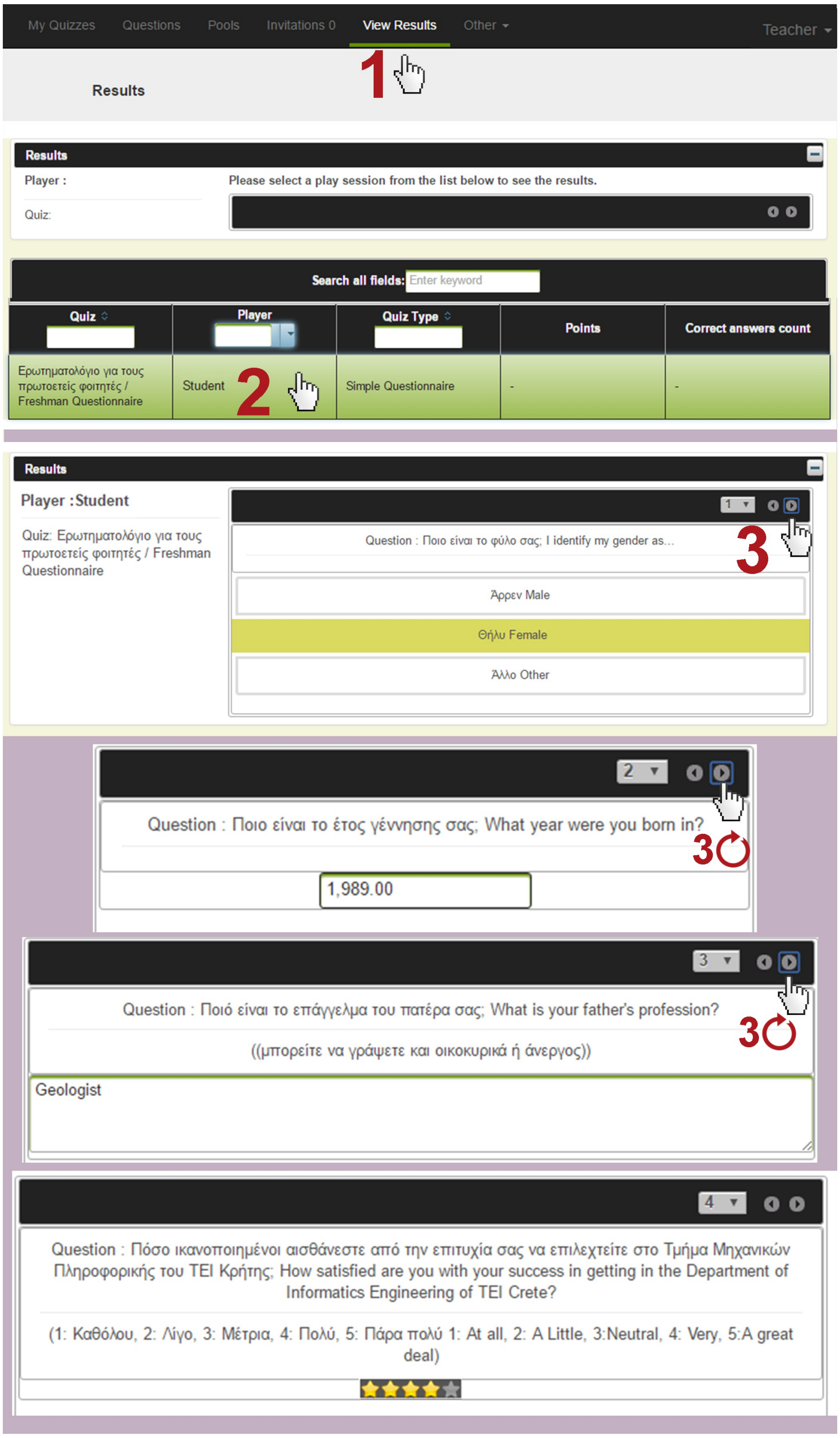
As usual, the player repeats *step 5* to go to the next and last question of this example. This question is of type *Rate*. When this question was created, the Teacher set the number of stars to 5. So now, the player has five stars to fill by clicking the desired amount. In this scenario, the player selects 4 out of 5 stars (step 8).

It is reminded that the total amount of questions in the questionnaire is 20 but only 4 of them are examined in this scenario. The player can browse through the questions by selecting τhe Next and Previous buttons. At any time, the player can submit the answers to the quiz by selecting *Submit* (step9).

By submitting the quiz, an outro screen appears with a *Return* option. The user can press it in order to close the game environment (step 10).

Back in the invitations page, we can see that the number in the *Invitation* menu and the *Awaiting* section are now zero. Instead, the *Completed(1)* section contains the quiz that was just taken.

View Results



Results

Results

Player :

Quiz:

Please select a play session from the list below to see the results.

Search all fields: Enter keyword

Quiz	Player	Quiz Type	Points	Correct answers count
Ερωτηματολόγιο για τους πρωτοετείς φοιτητές / Freshman Questionnaire	Student	Simple Questionnaire	-	-

Results

Player :Student

Quiz: Ερωτηματολόγιο για τους πρωτοετείς φοιτητές / Freshman Questionnaire

1

Question : Ποιο είναι το φύλο σας; I identify my gender as...

Άρρεν Male

Θήλυ Female

Άλλο Other

2

Question : Ποιο είναι το έτος γέννησης σας; What year were you born in?

1,989.00

3

Question : Ποιό είναι το επάγγελμα του πατέρα σας; What is your father's profession?

((μπορείτε να γράψετε και οικοκυρικά ή άνεργος))

Geologist

4

Question : Πόσο ικανοποιημένοι αισθάνεστε από την επιτυχία σας να επιλεχτείτε στο Τμήμα Μηχανικών Πληροφορικής του ΤΕΙ Κρήτης; How satisfied are you with your success in getting in the Department of Informatics Engineering of TEI Crete?

(1: Καθόλου, 2: Λίγο, 3: Μέτρια, 4: Πολύ, 5: Πάρα πολύ 1: At all, 2: A Little, 3:Neutral, 4: Very, 5:A great deal)

★★★★☆

Illustration 33: View Results

The scenario pictured on the left is from the point of view of the user *Teacher*. In this example, the user will view the answers given by the user *Student*, on the *Freshman Questionnaire* quiz.

First, the Teacher must be redirected to the results page by selecting the *View Results* option from the main menu (step 1).

This page is divided into two sections. The downer section contains a table with a list with all play-sessions associated with the logged-in user (quizzes that the logged user has created & played and quizzes that other users have played after an invitation from the logged user). Each row of the list shows information about the play-session (Quiz title, player's user-name, quiz type etc). The user can filter the list by quiz title or player and can also short it by quiz type. As the picture shows, in this example the list has only one row that represents the play-session of the user *Student* that took the quiz *Freshman Questionnaire*. To view the answers that the Student gave, the Teacher clicks the row of the table-list (step 2).

Now that a play-session is selected, the upper part of the page is updated and shows information about the Student's answers. On the right, the user can see the player's user name and quiz title and on the left, the first question and chosen answer is already shown. The Teacher can browse through the Students answers by selecting the Next-arrow (or previous-arrow) button (step 3).

4. Conclusions and Future Work

4.1 MyQuizDB Recap

This thesis has introduced an online quiz generator for educational purposes. After the registration, any user can create four types of quizzes:

- **Quiz with a basic points system:** The player can earn points from a question only when the given answer is completely correct (for example, in a case of a multiple choice question, the player must choose all the correct answers).
- **Quiz with penalty system:** The player earns points for each correct answer but looses points for each wrong choice.
- **Simple Questionnaire with no points:** This kind of quiz involves questions that happen for research or statistics reasons. There are not right or wrong answers and no earning or loosing points.
- **A rating system:** This is a classic system where the player can give a range type of answer by simply filling the rating star button.

The system also four types of questions as well. A quiz author can create and add questions

- **Multiple Choice:** The author creates the question and enters multiple possible answers. If the quiz type allows it, the user can mark each possible answer as correct or wrong and enter penalty or reward points accordingly.
- **Free Text:** This type of question allows the player to enter an answer during a play-session. If the quiz has a point's system, the author of the quiz can evaluate the player's answer as correct or wrong through a checkbox when viewing results.
- **Free Numeric:** Only numbers are accepted to this type of questions. The question creator can restrict the player's input by setting a minimum and maximum limitation.
- **Rate:** A rating question is answered by filling the star buttons. The author can define the amount of stars that will be given to the player (up to 10 stars).

A logged user can not only create but share quizzes as well. A quiz can be sent as an invitation to any other user of the system. Once the invitation has been sent, the receiver can view it at the **Invitations** page. When the receiver takes the quiz and submits the answers, the results will be sent back to the sender. All results can be reviewed at the **View Results** page.

Apart from quizzes, the system provides the possibility for creating question banks called **Pools**. The user can create create a Pool and add new questions directly to it. Questions from quizzes can also be added to any pool. Even during a play-session the player can add each question to one of the pools that belongs to him/her.

4.2 Future Work

There are many sections that have space to improve and one of them is the *User Management*. Giving the client the ability to create different types of accounts increases the target audience of the application by providing extra functionality for each user.

A second upgrade of significant importance would be the enrichment of *Question and Answer Management.* The types of questions and answers that MyQuizDB provides cover a big range of client demands but there are numerous extra features that could be added like image/ video/ file upload possibility as questions and answer types.

In conclusion, there are a lot of areas that can be improved or extended further. The work performed in this project provides basis for future development on quiz and education.

Bibliography

1: , , https://eclipse.org/
2: , , https://www.visual-paradigm.com/
3: , , http://www.adobe.com/products/photoshop.html
4: , , http://www.retrievalpractice.org/strategies/
5: QuizWorks, , https://www.onlinequizcreator.com/
6: Movement, QZZR, www.qzzr.com
7: Bluehouse Group, QuizBean, http://www.quizbean.com/home
8: investintech.com, What is a web application?, http://www.investintech.com/content/webapplication/
9: Tony Marston, What is the 3-Tier Architecture?, http://www.tonymarston.net/php-mysql/3-tier-architecture.html#what-3-tier-is-not
10: Mark Otto, Jacob Thornton, Bootstrap, http://getbootstrap.com/
11: Daniel Eden, Animate.css, https://daneden.github.io/animate.css/
12: PrimeTek, PrimeFaces, http://www.primefaces.org/
13: Oracle Corporation, JavaEE, http://www.oracle.com/technetwork/java/javaee/overview/index.html
14: , , http://hibernate.org/orm/
15: , Phpmyadmin, https://www.phpmyadmin.net/
16: Oracle Corporation, MySQL, https://www.mysql.com/
17: Wikipedia, Waterfall Model, https://en.wikipedia.org/wiki/Waterfall_model

Other References

http://ww2.kqed.org/mindshift/2014/10/16/studying-with-quizzes-helps-make-sure-the-material-sticks/
http://psych.wustl.edu/memory/research/
http://www.aboutintelligence.co.uk/intelligence-memory.html
http://allpsych.com/psychology101/memory/
https://en.wikipedia.org/wiki/Testing_effect
http://www.matlabtips.com/computer-vs-human-memor/
http://www.americanradioworks.org/segments/learning-to-love-tests/
https://www.theseus.fi/bitstream/handle/10024/94412/Final_Thesis_Anatolii_Shakhov.pdf?sequence=1
http://www.investintech.com/content/webapplication/
https://eclass.hua.gr/modules/document/file.php/DIT187/%CE%A0%CE%91%CE%A1%CE%9F%CE%A5%CE%A3%CE%99%CE%91%CE%A3%CE%95%CE%99%CE%A3/se02_requirements.pdf
http://searchsoftwarequality.techtarget.com/definition/software-requirements-specification
http://www.cs.ccsu.edu/~stan/classes/CS530/Notes14/02-SoftwareProcesses.html
http://www.icsd.aegean.gr/website_files/proptyxiako/16828056.pdf
https://en.wikipedia.org/wiki/Functional_requirement
http://www.sparxsystems.com/downloads/whitepapers/The_Use_Case_Model.pdf
http://www.icsd.aegean.gr/website_files/proptyxiako/736332428.pdf
http://www.sviluppoeconomico.gov.it/images/stories/garenato/IFB_3Parte
https://re-magazine.ireb.org/issues/2015-2-bridging-the-impossible/modeling-requirements-with-sysml/
http://shiro-takamura.deviantart.com/art/Curly-Brackets-Brush-Pack-300910997
http://coby17.deviantart.com/art/Arrows-Photoshop-Brushes-462955134
https://guidetogradschoolsurvival.wordpress.com/2011/04/15/how-to-write-future-workconclusions-2/
https://apothesis.lib.teicrete.gr/bitstream/handle/11713/3588/MarkopoulouMaria2007.pdf?sequence=1
https://apothesis.lib.teicrete.gr/bitstream/handle/11713/7739/KavvalouMaria2016.pdf?sequence=5&isAllowed=y
https://filmora.wondershare.net/