



Jakob Schweighofer, BSc

Development of a Quiz / (Self-) Assessment Tools and their Integration in Moodle

Master's Thesis

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Software Development and Business Management

submitted to

Graz University of Technology

Supervisor

Priv.-Doz. Dipl.-Ing. Dr.techn. Martin Ebner

ISDS - Institute for Interactive Systems and Data Science

Graz, July 2019

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Date

Signature

Abstract

E-Learning has become more popular in recent times and many organizations and universities use it as a key instrument in various teaching and training scenarios.

This master thesis gives a short overview of common assessment strategies in e-Learning systems and introduces a selection of popular examples. Furthermore, in the course of this thesis, a PHP-based quiz application was developed that can be integrated into Moodle by using the Learning Tools Interoperability protocol (LTI).

The PHP application is built to support programmable questions that can contain JavaScript and HTML code and random variables. Teachers are able to build interactive, randomized quizzes, in which the random variables can be assigned with complex mathematical functions. Additionally, the application provides a programmable grading mechanism. With this mechanism, it is possible for students to self-assess their performance, as well as for teachers to formally assess their students' learning success and send the results back to Moodle (or any other LTI compatible consumer application).

The features, as well as the implementation of the application, are described, followed by a discussion part and an outlook containing ideas for future extensions. A conclusion describes the insights that were gained over the course of the implementation.

Kurzfassung

E-Learning nimmt eine immer wichtigere Rolle in der heutigen Zeit ein und viele Universitäten und Betriebe nutzen e-Learning als zentrales Element in verschiedenen Lern- und Trainingsszenarien.

Diese Masterarbeit bietet einen kurzen Überblick über gängige Bewertungsstrategien in e-Learning-Systemen und stellt einige populäre Beispiele für solche Systeme vor. Weiters wurde im Rahmen dieser Arbeit eine PHP-basierte Quiz-Applikation entwickelt, die unter Verwendung des Learning Tools Interoperability (LTI) Protokolls in Moodle integriert werden kann.

Die PHP-Applikation unterstützt programmierbare Fragen, die JavaScript und HTML code, sowie zufällige Variablen enthalten können. Lehrende sind so dazu in der Lage, interaktive Quizze mit randomisierten Elementen zu erstellen, welche Werte aus komplexen mathematischen Funktionen beinhalten können. Die Applikation bietet einen programmierbaren Bewertungsmechanismus, der durch Lehrende dazu benutzt werden kann, um eine flexible Bewertung zu erzielen. Durch diesen Mechanismus können StudentInnen auch ohne Betreuung ihren Lernerfolg selbst bewerten und ebenso, wenn gewünscht, der/die Lehrende die Leistungen seiner/ihrer StudentInnen bewerten. Die berechneten Ergebnisse können an Moodle (oder andere LTI-kompatible "Consumer") zurückgesandt werden.

Die Funktionen der Applikation, sowie wichtige Punkte deren technischer Umsetzung werden in unterschiedlichen Kapiteln beschrieben. Es folgen ein Diskussionsteil, welcher kritische Punkte behandelt und ein Ausblick, der Ideen für zukünftige Erweiterungen beinhaltet. Abschließend werden einige Erkenntnisse, die bei der Entwicklung der Applikation gewonnen wurden, präsentiert.

Contents

| | |
|--|------------|
| Abstract | iii |
| 1 Introduction | 1 |
| 2 Background Information | 3 |
| 2.1 Educational Technology and e-Learning | 3 |
| 2.2 Online (Self-) Assessment | 4 |
| 2.2.1 Self-Assessment Studies | 6 |
| 2.3 Online Learning Systems | 7 |
| 2.3.1 Moodle | 8 |
| 2.3.2 edX - Online Courses | 9 |
| 2.3.3 iMoox | 11 |
| 2.3.4 MOOCs compared to LMS | 11 |
| 2.4 The Learning Tools Interoperability Protocol | 12 |
| 3 Development of a Quiz | 14 |
| 3.1 Goals | 15 |
| 3.2 User Perspective | 16 |
| 3.2.1 Starting the Application | 16 |
| 3.2.2 Navigation | 19 |
| 3.2.3 Creating Questions | 21 |
| 3.2.4 Creating Quizzes | 34 |
| 3.2.5 Taking a Quiz | 41 |
| 3.2.6 Showing Solutions | 45 |
| 3.2.7 Processing the Quiz Results | 45 |
| 3.2.8 Administering Users, Roles and Permissions | 49 |
| 3.3 Technical Perspective | 52 |
| 3.3.1 Zend Framework | 52 |
| 3.3.2 The IMS Global LTI Framework and User Authentication | 53 |

Contents

| | | |
|----------|--|-----------|
| 3.3.3 | The LTI Outcome Service - The Grade Book | 56 |
| 3.3.4 | Testing LTI Applications | 57 |
| 3.3.5 | User Interface: Bootstrap, jQuery and jQueryUI | 57 |
| 3.3.6 | Database | 58 |
| 3.3.7 | Security Considerations | 61 |
| 4 | Discussion | 65 |
| 5 | Outlook | 67 |
| 5.1 | Gamification | 67 |
| 5.2 | Learning Analytics | 68 |
| 5.3 | Evaluation Study | 68 |
| 6 | Conclusion | 69 |
| | Glossary | 72 |
| | Appendices | 74 |
| A.1 | Example LTI Launch Request | 75 |
| A.2 | Outcome Service - Replace Result Request XML | 77 |
| | Bibliography | 79 |

List of Figures

| | | |
|------|---|----|
| 2.1 | The e-Learning process as depicted in Barbosa and Garcia, 2005, p.1 | 5 |
| 3.1 | The home screen as seen by instructors | 18 |
| 3.2 | Login screen for administrators | 19 |
| 3.3 | The top menu as seen by administrators | 20 |
| 3.4 | Breadcrumb path when editing questions | 21 |
| 3.5 | List of questions | 22 |
| 3.6 | Top section of the new questions form with example question text | 23 |
| 3.7 | Question view for an example question | 25 |
| 3.8 | Question preview parameters with their solutions below the question text in the question view | 27 |
| 3.9 | Question preview parameters and solutions are generated on a dedicated page | 28 |
| 3.10 | Question solution guide for the example question | 32 |
| 3.11 | The list of quizzes with indicators for live and ended quizzes | 34 |
| 3.12 | Creating a new quiz | 35 |
| 3.13 | The quiz preview page, showing a live quiz | 37 |
| 3.14 | The question assignment page | 39 |
| 3.15 | Collapsible list of generated quiz parameters | 40 |
| 3.16 | Example input made on the quiz preview page to test the outcome of the answer | 41 |
| 3.17 | The quiz start page in the resume state | 42 |
| 3.18 | Severity levels of buttons | 43 |
| 3.19 | The quiz flow | 44 |
| 3.20 | The solution page | 46 |
| 3.21 | The quiz grading page | 47 |

List of Figures

| | | |
|------|--|----|
| 3.22 | The question grade review page, comparing students' answers with correct solutions | 49 |
| 3.23 | Administrator role, showing inherited permissions and the LTI name | 51 |
| 3.24 | The LTI consumer emulator by the ceLTic project | 58 |
| 3.25 | Database model of the quiz application | 59 |

1 Introduction

"One child, one teacher, one book and one pen can change the world."
(Malala Yousafzai, 2013, p.596)

This quote from the female peace Nobel prize winner Malala Yousafzai, emphasizes the tremendous importance of education and the availability of educational resources. No one can ever know, which inventions and insights have already been lost due to the absence of proper education in the right place, at the right time.

Since the rise of personal computers and the internet, these two technologies have transformed nearly every aspect of life, one of which is learning.

The importance of e-Learning technology has steadily grown since then and many universities use e-Learning as a key instrument in their education programs (Harandi, 2015).

Not only schools and universities, but also businesses all around the world start to utilize the powerful tools provided by modern e-Learning systems. Most of these applications have (at least to some degree) implemented assessment or self-assessment methods, which are essential tools of educational processes (Stan and Manea, 2015).

The reason why there is a trend towards the rising use of e-Learning becomes clear when looking at, for example, the study conducted by Ćukušić, Garača, and Jadrić (2014). It revealed a statistical significance showing that the final exam pass rate increases if self-assessment is practiced during the semester. Another interesting study was performed by Harandi (2015) and showed a significant relationship between e-Learning use of e-Learning and increased student motivation. Section 2.2.1 mentions some studies that were performed in the sector of e-Learning.

1 Introduction

This thesis is divided into two major parts, whereas one is of theoretical and the other of practical nature. The theoretical part gives an overview of common terminologies in the context of e-Learning and shortly introduces some e-Learning applications, with a focus on their self-assessment capabilities. In the course of the practical part of this thesis, a (self-) assessment tool was created that can be integrated into Moodle. The quiz application enables teachers to create quizzes with content that they can program in JavaScript and HTML, as well as common text. Even though Moodle already offers the possibility to create quizzes, it lacks the needed flexibility to define variables and solutions with complex mathematical functions that are for example included in common JavaScript mathematics libraries. The quiz application tries to solve this shortcoming with providing questions that can utilize the full power of JavaScript and HTML. The primary purpose of the application is to implement a process for students to self-assess their learning progress during a semester. In addition to the self-assessment, it is also possible for a teacher to formally assess the student participation in the quizzes and use the software as an assessment tool. The quiz application is compatible with the LTI protocol and can be launched from Learning Management Systems, such as Moodle.

The detailed description of how this application was made and how the integration into Moodle was achieved, can be seen in the chapter 3 and its subsections 3.2, which describes the features and usage of the application and 3.3, which describes the technologies used to build the application.

Finally, the thesis will present an outlook on how the quiz application could be extended and analyzed in the future, followed by a discussion part which describes the problems that were faced and are present in the implementation of the quiz application. The last chapter consists of a conclusion about the creation of LTI compatible (self-) assessment tools.

2 Background Information

In the subsequent chapters, some terms are used that require a precursory explanation. This section contains background information about e-Learning and its specific processes assessment and self-assessment and gives an overview of common terms, used in the context of e-Learning and online assessment. In addition, some of the most popular and successful e-Learning technologies are described.

2.1 Educational Technology and e-Learning

E-Learning has become more important ever since the invention of the internet and even before, different ways of distance learning have been existing that range back into the 1980s and earlier (Harasim, 2000). Even though e-Learning has found broad adoption in many ways, there is no single definition of what e-Learning is and multiple descriptions exist.

Educational Technology is a term related to e-Learning. According to the AECT (Association for Educational Communications & Technology) the definitions is as follows:

“Educational Technology is the study and ethical practice of facilitating learning and improving performance by creating, using and managing appropriate technological processes and resources.” (Richey, Silber, and Ely, 2008, p.1)

As per this definition, the following terms mentioned in 2.2 and 2.3, can be considered as subjects of educational technology.

2.2 Online (Self-) Assessment

This section elaborates on the forms of online assessment and self-assessment. In the educational context, assessment is a general term that can be understood in different ways. Thus, it is necessary to provide clarification on what form of assessment is referred to.

By "assessment" one can mean that students assess the learning material provided by a teacher, or it could also mean that the teacher holds a final exam and assesses the students' achievements with the outcome of the exam. Additionally, teachers could adapt their teaching strategy by analyzing the exam results and learn on their end by interpreting and assessing their students' grades. Yet another form of assessment is self-assessment. In this case, a learner assesses his or her own learning progress.

There is a general consensus about the two main purposes of assessment, which are a summative and a formative purpose. The first is a certification purpose and the latter is described as a learning purpose. (as cited in Liu and Carless, 2006)

Barbosa and Garcia (2005) emphasize that the assessment process should not be seen as an isolated subject without taking the whole e-Learning process into consideration. See figure 2.1 for a basic e-Learning process with assessment, as described by Barbosa and Garcia. When speaking about educational assessment, one should always keep the frame of reference and purpose of the discussed assessment strategy in mind. To elevate this recommendation into the e-Learning context, this could mean that when designing a system that must perform such assessment activities, one should always keep the whole process of adaption of learning material and propagation of feedback to both, learners and teachers in mind.

While the term assessment is generally used to describe a form of grading, ranking or evaluation of performance, there is a sub-category that is called peer assessment, in which students grade the outcomes of their colleagues and the teacher shares his or her responsibility with the students (Liu and Carless, 2006).

Self-regulation and self-regulated learning are terms, that can be closely associated with self-assessment, as in most descriptions, self-assessment is

2 Background Information

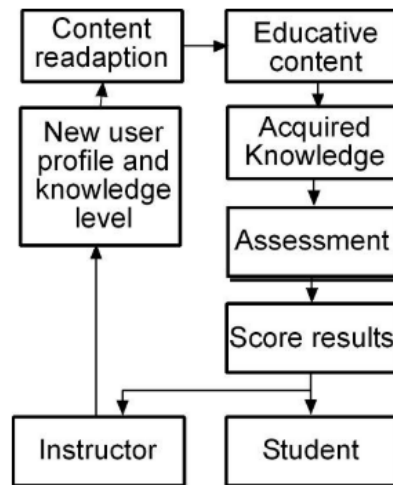


FIGURE I
ASSESSMENT IN THE E-LEARNING PROCESS

Figure 2.1: The e-Learning process as depicted in Barbosa and Garcia, 2005, p.1

a part of the self-regulated learning process. Self-regulated learning can be divided into self-regulation processes and self-regulation strategies to improve these processes. The perception of self-efficacy is also part of the self-regulation process and utilized in a feedback loop, that is described in most definitions of self-regulated learning (Zimmerman, 1990).

The model which is shown in figure 2.1 also represents a cyclic model of the e-Learning process. The feedback-loop that is part of the self-regulated learning process can be incorporated into an e-Learning application with the utilization of retries and displaying of the correctness of the results between the single tries.

“Retries are the ability to allow multiple attempts to answer a question. Obviously, this ability is of great importance for self-assessment, since it may be useful to improve the knowledge of the student while reducing the need to provide feedback and/or tutoring.” (Valenti, Cucchiarelli, and Panti, 2002, p.166)

The quiz application that is introduced in the later chapters of this thesis also

2 Background Information

implements a retry mechanism. Together with a clear assessment criterion that is defined by the teacher, students should be able to successfully self-assess their performance and improve it over the course of a semester. For teachers, the application should be a useful and particularly time-saving tool to assess their students' learning progress.

The mentioned assessment methods are all directed towards the evaluation of the learner's accomplishment and are a part of most e-Learning systems, even if not mentioned explicitly. Along the possibility to provide an online course with retries would qualify an application as a self-assessment tool. All of the e-Learning Systems that are introduced in section 2.3, have implemented self-assessment and peer assessment, besides the classical summative assessment, as evaluation methods for their course content.

Formative assessment was mentioned earlier in this section. It is described as an assessment process that is intended to improve the learning process of students by generating feedback for the teacher (Nicol and MacFarlane-Dick, 2006). Considering the purpose of formative assessment, the use of online (self-) assessment software can also be seen as a great opportunity to improve teaching strategies and ultimately improve the learning process of students, as the data gathered by such systems can be easily retrieved and analyzed.

2.2.1 Self-Assessment Studies

To get a better understanding of how especially self-assessment works in practice, some studies were selected to gain knowledge about the background of self-assessment among students. Amo and Jareño (2011) have conducted a study, that compares self- and peer assessment results, compared to their teachers' assessment results over the course of three years among Spanish students. The results indicated, that the self- and peer assessments of first-year students, were more likely to overestimate their performances, compared to the teachers' assessment. In the third year, the students' assessments tended to be rather underestimating their performance compared to the teachers' assessment, which was in no case lower than the student's perception of their success.

2 Background Information

This is interesting because one could derive from these results, that self-assessment is better suited, or at least more precise, for a more advanced audience.

These results are strengthened by the study of Stan and Manea (2015), which investigated the relationship between assessment and self-assessment. The study included 121 students in the field of primary and preschool pedagogy. The results indicate that the majority of participants tended to underestimate their performance.

Ćukušić, Garača, and Jadrić (2014) have performed a study that monitored the pass rates on final end-semester exams. The team chose one first-year undergraduate university course and provided self-assessment tests in the form of Moodle quizzes over the course of three years in a hybrid model (e-Learning paired with on-campus classes). In the second year a correlation between the achieved points of the self-assessment tests, half-semester and final tests, became significant. When the overall results of the monitored course were compared to other courses in the same time frame, the results showed, that the pass rate of the course which implemented the self-assessment strategy, was significantly higher than those of the ones without such strategies.

Another positive finding, which does not specifically refer to self-assessment, but to e-Learning in general, is presented by Harandi (2015), who carried out a study on the Tehran Alzahra University with 140 probands. Harandi provided a questionnaire that intended to evaluate if there is a correlation between the use of e-Learning technologies and the student's motivation and showed a correlation between the two.

Generally speaking, the use of e-Learning and self-assessment tools, seems to have positive effects on the learning experience and success rates of the learners.

2.3 Online Learning Systems

Many terms for Virtual Learning Environments exist. Some of which are, Web-Based-Trainings, Learning Management Systems, E-Portfolio Systems,

2 Background Information

MOOC Platforms, Personal Learning Environments and Immersive Learning Environments. The definition of these environments is hard to isolate and the e-Learning landscape is continuously evolving (Ebner, 2019).

This section does not try to give a comprehensive overview of existing forms of Virtual Learning Environments but investigates a small selection of popular systems in regard to their assessment strategies. This should deliver a brief impression of common assessment and self-assessment implementations in real-world examples.

2.3.1 Moodle

One of the most popular and well-established examples of learning management systems is called Moodle. The PHP-based open-source application was first published on 20 August 2002 and has been continuously improved since then. At the time of writing this thesis, the current version of Moodle is 3.6.3 with further releases already planned.

Moodle has many features that set standards for modern Learning Management Systems and can be visually adapted to blend into any custom branding by using themes.

Since version 2.2, Moodle has utilized the Learning Tools Interoperability (LTI) protocol (described in 2.4) which enables it to launch other learning systems as external tools and adds to its versatility. Since version 3.1 it has been possible to provide Moodle course content to external users, who can launch the courses with the use of the LTI protocol. The quiz application that was developed in the course of this thesis, makes use of the LTI capabilities of Moodle and can be integrated as external tool that launches via the LTI protocol.

Moodle also offers the possibility to create quizzes. There are many options that can be set for the Moodle quizzes, some of which are:

- Custom quiz introductions
- Complex time restrictions (time limits and time between tries)
- Various grading methods such as "Highest Grade" or "Last Grade"
- Retries / attempts

2 Background Information

It is possible to create questions based on a variety of question types that can also be supplemented by freely available question type Add-Ons¹. There are question types that already allow the creation of dynamic questions, containing randomized variables in the text. However, none of these question types offers the possibility to add complex functions for generating the random variables or evaluate mathematical expressions in answers.

Besides the classical summative assessment, Moodle also offers the possibility to provide self-assessed courses by enabling retries. Furthermore, Moodle offers workshops², which are peer-assessed courses that are based on the collaboration of students.

2.3.2 edX - Online Courses

The edX online course platform³ is a declared non-profit provider of so-called MOOCs (also explained in 2.3.4), that was founded in 2012 as a joint project by the Massachusetts Institute of Technology and the Harvard University.

Initially, edX was designed to be an open and completely tuition-free source of quality learning material but recently the announcement was made (see Agarwal, 2018), that there will be a paid and a free license, that differ in the quality of the provided service. Graded assessments as well as certifications and full course access, can only be consumed with the paid version from now on.

edX differentiates the free and paid version by calling them as different "tracks". The paid version is labeled the "verified track" and the free version is labeled the "audit track".

In addition to the general online courses, edX offers the e-Learning platform used for their online course system to the public. It can be acquired as a

¹Moodle Question Type add-ons <https://moodle.org/plugins/browse.php?list=category&id=29> 09.07.2019

²Moodle Workshop Module https://docs.moodle.org/24/en/Workshop_module 10.07.2019

³edX Online Courses <https://www.edx.org> 28.06.2019

2 Background Information

self-hosted, free variant and as a fully-managed variant, where depending on the third-party hosting service, additional fees incur. This product is called Open edX⁴.

If the amount of students is large enough, grading cannot be done manually anymore and therefore MOOCs must incorporate some form of automated grading process, or rely on peer and self-assessment strategies. The edX platform has various grading concepts depending on the course and incorporates self-, peer and summative assessment methods.

One interesting mode of operation for grading, that is described in the documentation, is that to successfully finish some courses, the student has to peer-assess a given number of other students. This is a viable solution for large amounts of participants but, depending on the overall knowledge of the students, it can be imprecise and prone to errors.

Courses on the edX platform can be either free, or generally require payment to attend the course. If a free course is chosen, no entry fees are defined and the participant still has the opportunity to choose the verified track or the audit track for a course. The price for the verified track is around 49 USD and can vary between the courses. The verified track includes assessment of the students' achievements by a teacher in most cases. If a course generally requires payment, the prices can be as high as 1300 USD. The prices that are described in this paragraph might not be representative for the edX platform. Only a small, random selection of free and paid courses was investigated.

If participants choose the free trail for available courses, the applied assessment method is self-assessment. Stated in the edX documentation about checking ones learning progress⁵, the students are provided an overview over their progression and received points and can self-assess their achievement.

⁴Open edX <https://open.edx.org> 09.07.2019

⁵Checking Learning Progress in edX https://edx.readthedocs.io/projects/edx-guide-for-students/en/latest/SFD_check_progress.html 10.07.2019

2 Background Information

2.3.3 iMoox

The iMoox platform was founded in the year of 2013 as a common project of the University of Graz and the Graz University of Technology. It is Austria's first MOOC platform⁶ and all courses offered on the iMoox platform are tuition-free. Courses provide interactive learning materials containing videos and quizzes. The iMoox platform is based on Moodle.

The assessment method in iMoox is strongly dependent on the course. Self-assessed quizzes are common, but also discussion forums are utilized by some of the courses, which could be seen as a form of peer assessment. A benefit of discussion forums is that, not only all students can profit from reading in the forums, but also teachers can interact with the participants of the courses and provide guidance.

The use of discussion forums in e-Learning environments, specifically in MOOCs can increase the connectedness of students, which can decrease the overall drop-out rates in MOOCs (Khalil and Ebner, 2016).

2.3.4 MOOCs compared to LMS

In the former sections, two MOOC systems and one Learning Management System were introduced. To summarize the differences and further strengthen the reader's knowledge about these two systems, a short comparison follows based on the overview given by Ebner (2019).

MOOCs are a newer approach and trend in e-Learning systems compared to the more traditional Learning Management Systems.

Where MOOCs mainly focus on providing open online courses and are very centered on the courses, Learning Management Systems are more focused on the teachers and supporting them in managing their learning resources. MOOCs are intended to be served to a large audience and LMS tend to include features to manage users and user groups. Furthermore,

⁶About iMoox <https://imoox.at/mooc/theme/imoox/views/about.php?lang=en>
10.07.2019

2 Background Information

LMS usually include options for different types of online assessment (Ebner, 2019).

A survey, that was carried out on the Australian National University, for a popular series of MOOCs, yielded the result that 87% of the participants found that they gained as much or more from MOOCs as compared to their on-campus learning experiences (Palmer, 2015).

How exactly the survey was conducted, is not clear and the comparison between MOOCs and on-campus courses could be arguably hard and very subjective. However, the user experience of well-implemented MOOCs seems to be accepted well and generally liked by the users.

To also include a critical voice, contrary to many proponents of MOOCs, Vardi (2012) has some interesting critical economic thoughts about the topic. In his article, he refers to the financial situation of colleges in the United States of America after the financial crisis and the following recession. Vardi also criticized the overall pedagogic quality of university education. He theorizes that due to the high tuition fees and lesser chances on the job market (which also suffers from recession), it could be very attractive to not take a student loan and matriculate at a university but just enroll in MOOCs, causing further budget cuts justified by lesser cost due to reduced numbers of students. (Vardi, 2012)

2.4 The Learning Tools Interoperability Protocol

The Learning Tools Interoperability Protocol (LTI) was developed by the IMS Global Learning Consortium⁷ and its purpose is to standardize the communication between online learning systems. The one side of the connection is called the tool consumer and the other side is called the tool provider.

The **tool consumer** can be generally considered as an arbitrary Learning Management System (LMS), which can be of any shape, design and mostly any technology, as long as it complies with the LTI specifications. Moodle is

⁷IMS Global Homepage <http://www.imsglobal.org> 29.06.2019

2 Background Information

an example of an LMS that can act as a tool consumer when external tools are included as course material.

The **tool provider** on the other hand, is a separated instance of educational software, which is launched by the tool consumer and can be anything from a link to a website with information to a comprehensive online course system itself. In the case of Moodle, it is even possible to launch other Moodle resources from Moodle, so the borders between tool consumer and tool provider can be blurred in some instances.

The LTI protocol uses a set of defined actions/requests, that are sent from the tool consumer to the tool provider. The tool provider must validate and implement it's reactions to those messages. A detailed description of how an implementation of this communication looks is provided in chapter 3.

3 Development of a Quiz

The quiz application that was developed in the course of this master thesis, was made to support physics lectures, held on the Graz, University of Technology. The lecturer of these physics courses desires to support the students with short virtual quizzes during the semester and the genuine idea for this application does not originate from the author of this thesis.

The application tries to minimize the effort for the teaching personnel to create educational resources and grade the student homework, while providing a successful learning experience for the students, to aid them with passing the final exams at the end of the semester.

The application is designed to give maximal freedom in crafting the questions to the creator of the quiz. It provides support for programmable questions containing HTML and JavaScript code, which enables the quiz creators to build interactive questions, that can be different for every student.

Despite the difficulty to provide the flexibility needed by a programmer, while normalizing the code via a graphical user interface, one of the main problems was managing variables for the questions. The solution to this problem is that all random variables must be defined at the time of question creation and then be stored in the database so that they can be efficiently retrieved later on when students take a quiz. This also avoids the exposure of JavaScript code that should be private and not reveal the concrete solutions to the problems, if the teacher does not specifically decide to show them.

The default nature of JavaScript is to be executed on the client-side. It leads to the problem that, if the random variables that are used in the quiz are not precalculated, the students would be able to see the functions used to create the values and their solutions. Therefore all variables that are used

3 Development of a Quiz

in the quizzes, as well as the solutions to the problem equipped with these values, are generated in the quiz creation process and persistently stored in a database. Many design choices revolved around this circumstance and will be explained in detail in the following sections 3.2 and 3.3.

The main features of the application are:

- Create questions that can contain JavaScript and HTML
- Create quizzes
- The possibility to launch quizzes with an LTI Launch Request from Moodle
- An interface to efficiently grade the results of the quizzes
- The possibility to send the grades back to the LTI consumer that launched the quiz

This section is divided into two major parts: The first part describes the application from a user's perspective and explains the major functionality and usage. The second part elaborates the technologies that were used, the structure and the composition of the application.

3.1 Goals

Time matters when creating exams or homework for hundreds of students while having limited personal resources. The primary focus of the application was to provide an easy-to-use web application that enables the lecturers and assistants to help the students with a (self-) assessment tool, to support the student learning process while using a minimum of human resources.

The goals of the application were:

- Time-efficient quiz and question preparation
- Reusability of learning material
- The freedom of using JavaScript and HTML when creating questions
- Minimal effort for students to find and launch a quiz
- Track the student success by grading their results
- Enable the use of the application as an unsupervised self-assessment tool

3 Development of a Quiz

- Enable the use of the application as a supervised assessment tool

3.2 User Perspective

This section describes the functionality of the application from a user's point of view. After reading this section, the reader should have a profound understanding of the features of the quiz application and should be able to use it. In the subsequent section 3.3, a more detailed technical description follows, which explains how the application was made and which technologies were used.

The term "user" is defined as, one or multiple of the three stakeholder groups of the application: Teachers, Students and Administrators. Each of them has access to different features of the application and depending on their role, a completely different view of the application.

3.2.1 Starting the Application

There are multiple ways to start the application once it has been set up, depending on the person who is starting it. Lecturers, Administrators and Students should all be defined in the Moodle (or other LTI compatible LMS) instance that is used to manage courses and the access to them. Defined by the LTI standard, there exist numerous standard roles that should lead to a different visualization of the application. To start LTI providers from Moodle, the application is embedded as so-called "External Tool" activity in the consuming Moodle environment and the entry point is defined as:

```
https://[public-moodle-webroot]/lti/connect
```

Depending on the web server that is used to install the quiz application, the launch URL must be adapted accordingly. The LTI role mentioned above is very important for the application, as it separates access to the functionalities of the application by the given role with built-in role-based access control, which is also commonly abbreviated as RBAC.

3 Development of a Quiz

The role-based access control is based on the examples in the open-source book Oleg (2018) and described in detail in section 3. What is important, is to know that different roles are defined in the quiz application, which are mapped to the standard LTI roles. This means that when the LTI request validation is complete, typically the launched application establishes a session for the user. In the case of this application, the user is logged in and the user entry assigned to the user context is defined as one of the following:

- Administrator
- Instructor
- Learner

Case 1: Launching the Application as Lecturer/Tutor/Instructor

In this case, the launch URL of the quiz application is called by the LTI consumer (Moodle in this case) without the custom parameter "launch_key" (explained in 3.2.4) and with the role "Instructor" in the roles array contained in the LTI launch request.

If all of the above is the case and the LTI authentication is successful, the user is redirected to the application home page, which looks as shown in figure 3.1.

The home page contains a short description for the main features of the application in the order of a usual workflow, from left, to right.

The top menu is only visible for administrators and instructors, where only admins have access to the "Users" menu item and its sub-items users, permissions and roles.

If the application is started with a quiz link, containing the launch key custom parameter, the quiz starts as it would for a student. That means that instructors can also take quizzes and are not limited solely to the instructor functionality.

3 Development of a Quiz

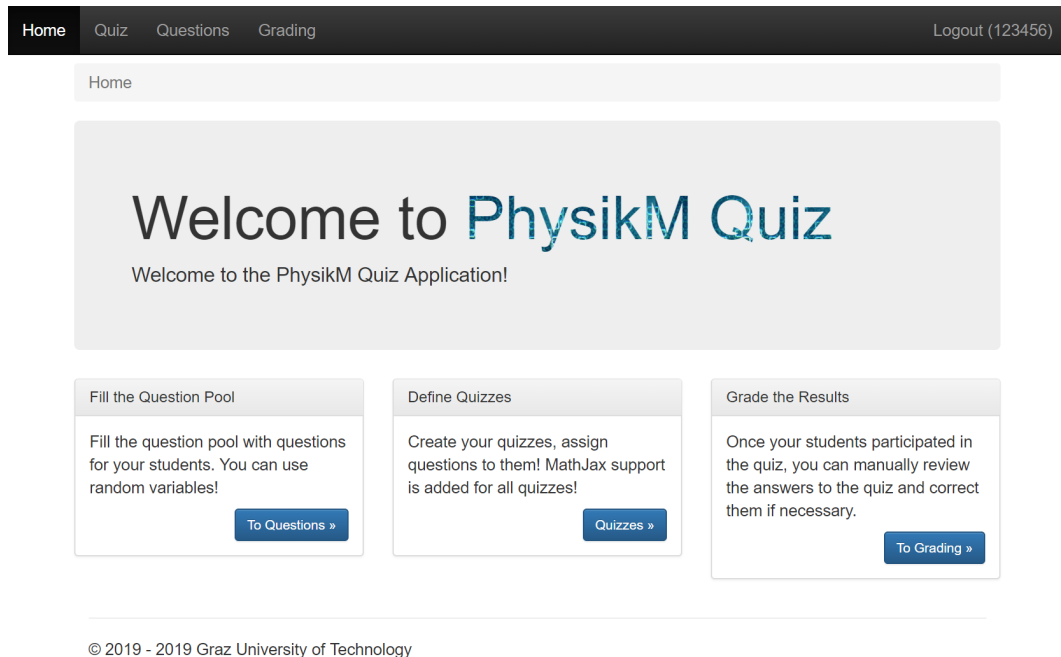


Figure 3.1: The home screen as seen by instructors

Case 2: Launching a Quiz as Student

The launch process for the students, in this context also referred to as “Learners”, which is the name of the according LTI role, is the same as for instructors. They click on the external tool activity that is defined in the LTI consumer and are redirected to the quiz application.

The LTI protocol offers the option to provide a custom parameter with the launch request. This custom parameter is used to specify which quiz will be launched once the user authentication is complete.

To specify the target quiz, the custom parameter must be named “launch_key” and must contain a valid launch key value. How this value can be tied to a quiz is explained in section 3.2.4.

If a quiz with the given launch key is found, the quiz starts. This is described in section 3.2.5.

3 Development of a Quiz

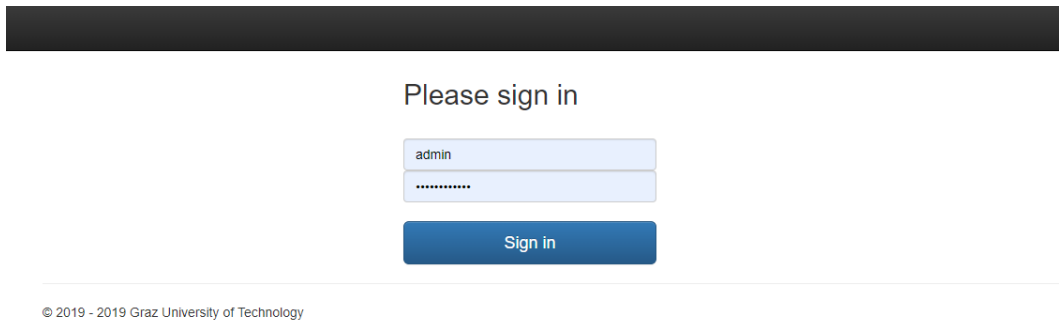


Figure 3.2: Login screen for administrators

Case 3: Launching the Application directly as Administrator

The application has a login page (Figure 3.2) which can be used by non-LTI users and is designed for administrators. These can log in and access the application without prior authentication through the LTI protocol.

The only two differences between administrators and instructors are that administrators have an additional menu item available in the top menu bar, where they can access users, roles and permissions for the application and that instructors cannot log in without LTI authentication. Changes in the user sections can be critical, potentially break the application and must therefore be handled with care. The administrator role is designed for technical support with knowledge and understanding about the foundations of the quiz application and not for regular users.

3.2.2 Navigation

The application navigation makes use of two very common elements in modern web sites. A top menu bar as primary navigation and breadcrumbs as secondary navigation.

3 Development of a Quiz

Primary Navigation - The Top Menu

Depending on the role of the current user he or she has access to different elements of the top menu. Figure 3.3 gives a view on the full menu from an administrator's perspective.

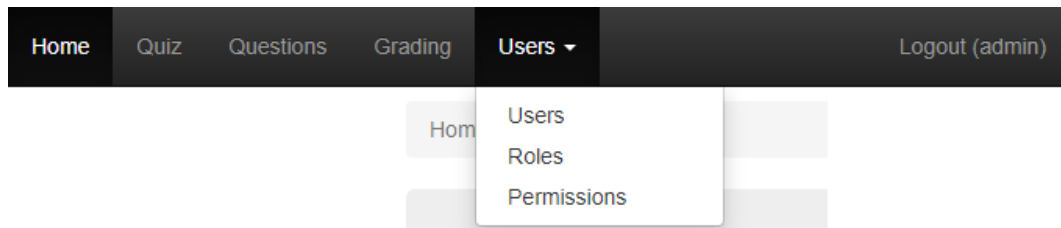


Figure 3.3: The top menu as seen by administrators

- Students: No top menu items
- Instructors: Access to Home, Quiz, Questions, Grading
- Administrators: Access to Home, Quiz, Questions, Grading, Users

Clicking on a menu item usually leads to an overview page that contains a list of all entries of the specified kind. For example, when instructors click on the "Quiz" menu item, they are redirected to a page, containing a paginated list of quizzes from where they can perform all quiz-related tasks.

Secondary Navigation - Breadcrumbs

The use of breadcrumbs has steadily increased over time and users became so used to them, that they sometimes criticize missing breadcrumbs as a missing feature (Nielsen, 2007).

For this reason and to generally ease the use of the quiz application, breadcrumbs were included. Figure 3.4 displays the breadcrumb path when an instructor or admin is editing a question.

3 Development of a Quiz



Home / Questions / View Question 23 / Edit Question 23

Figure 3.4: Breadcrumb path when editing questions

3.2.3 Creating Questions

Every quiz consists of questions.

The idea of the questions is to guarantee the reusability of the learning material and save time for the teachers in preparing them. A quiz can consist of just one or multiple questions, which have to be created beforehand. The questions can be reused in an unlimited number of quizzes and should satisfy the achievement of the goal of reusability of learning materials, described in 3.1. To create a new question an instructor or administrator has to navigate to the menu item "Questions", where they see a list of current questions, sorted by creation date in descending order (Figure 3.5). Learners do not have access to this feature. The list also provides a quick display of the points and time estimate for the question. The page size for the list is set to ten questions per page and the pagination control is attached at the top and bottom of the list.

Located on the left side, above the overview list, is a link with the label "Add New Question" that leads to the form for the question creation, also visible in Figure 3.6.

Every field on the question form is important, but there are six input areas that are special because they produce and influence the question content. These question elements generate the actual question content and therefore are called "Content Generators" or "Content Generator Functions" in the following sections.

The six content generators are:

- The Question Text: $T(Q)$
- The User Input Area: U
- The Question Parameter Generator Function: Q
- The Solution Generator Function: $S(Q)$
- The Grading Function: $G(U,S)$

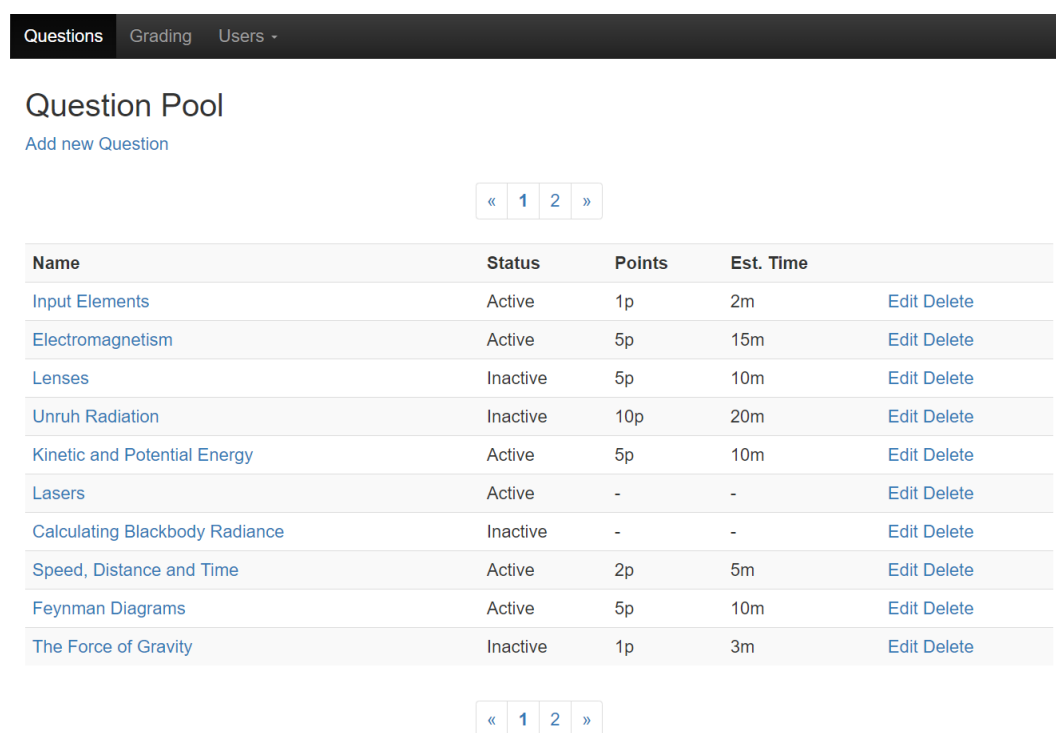
3 Development of a Quiz

- The Solution Guide $SG(Q)$

The notation in the bullet list gives an indication of what inputs are available for the respective content generator element. For example, the question text T has all values of Q at hand and can use them in the text. The grading function G has two inputs available, which are needed because the purpose of the function G is to compare the user answers with the correct solutions and calculate a grade as a result of the comparison. A detailed description of the content generators follows.

The Question Text: $T(Q)$

The main input element on the new question form is the "Question Text". This text area input field is intended to serve primarily as input for HTML



Questions Grading Users ▾

Question Pool

[Add new Question](#)

« 1 2 »

| Name | Status | Points | Est. Time | |
|--|----------|--------|-----------|---|
| Input Elements | Active | 1p | 2m | Edit Delete |
| Electromagnetism | Active | 5p | 15m | Edit Delete |
| Lenses | Inactive | 5p | 10m | Edit Delete |
| Unruh Radiation | Inactive | 10p | 20m | Edit Delete |
| Kinetic and Potential Energy | Active | 5p | 10m | Edit Delete |
| Lasers | Active | - | - | Edit Delete |
| Calculating Blackbody Radiance | Inactive | - | - | Edit Delete |
| Speed, Distance and Time | Active | 2p | 5m | Edit Delete |
| Feynman Diagrams | Active | 5p | 10m | Edit Delete |
| The Force of Gravity | Inactive | 1p | 3m | Edit Delete |

« 1 2 »

Figure 3.5: List of questions

3 Development of a Quiz

The screenshot shows a web interface for creating a new question. At the top, there is a navigation bar with 'Quiz', 'Questions', 'Grading', and 'Users' menus, and a 'Logout' link. Below the navigation bar is a breadcrumb trail: 'Home / Questions / New Question'. The main heading is 'New Question'. The form consists of several sections:

- Title:** A text input field containing 'Speed, Distance and Time (Avg)'.
- Question Text:** A large text area containing:

```
How fast is a car driving (in km/h) when it travels {{carDistance}} km in {{carTime}} hours?  
  
<br/><br/>  
  
These students move at different speeds:<br/>  
<ul>  
<script>  
  for (i in {{students}}) {  
    document.write("<li>" + {{students}}[i] + " moves at the speed of " + {{studentSpeeds}}[i] + " km/h</li>");  
  }  
</script>  
</ul>
```
- Number of Preview Parameters:** A text input field containing '3'.
- Parameter Function Q:** A text area containing:

```
var Q = {  
  carDistance : Math.floor(1 + Math.random() * Math.floor(15)),
```

Figure 3.6: Top section of the new questions form with example question text

and JavaScript code, but can also be used to create pure text questions. The stored question text will be printed exactly as it is in its quiz environment, containing all HTML and script tags. Only one element from the question text is replaced: the question variables.

It is not necessary, but possible to use variables within the text, which serve as randomization elements to make every quiz trial unique and more challenging. The variables are generated by a JavaScript function that can also be defined in the question form and will be executed after the creation of the question. Question variables must be marked with a “{{” symbol at the beginning and a “}}” symbol at the end of the variable name for example “{{speed}}”. An example of how a question text with variables could look is shown in listing 3.1

Every time the question text is displayed in a quiz or the question preview page, the variable placeholders are replaced with actual values.

For the variables to work within the question, the next step is to define how the values for them are created. Therefore, in addition to the main question

3 Development of a Quiz

text input field, there are four other important fields, which are involved in producing the final question as seen in the question view (figure 3.7) and are strongly dependent on each other.

The Question Parameter Function Q

This function is used to generate the values for the variables and must be written in JavaScript. It is called "Question Parameter Function Q" or "Parameter Function" in the following sections and it can also be seen in figure 3.6, below the question text input area. The JavaScript object that is produced by the parameter function, is called "Q". A defined set of instances of Q is generated for each question and each instance can have multiple variables enclosed in the object. The set size is defined by the number of preview parameters in the question preview and the number of quiz parameters in the case of a quiz.

```
1 How fast is a car driving (in km/h) when it travels
2 {{carDistance}} km in {{carTime}} hours?
3
4 <br/><br/>
5
6 These students move at different speeds:<br/>
7
8 <ul>
9 <script>
10     for (i in {{students}}) {
11         document.write("<li>" + {{students}}[i] +
12             " moves at the speed of " + {{studentSpeeds}}[i] +
13             " km/h</li>")
14     };
15 }
16 </script>
17 </ul>
18
19 <br/>
20
21 Calculate the average speed of the group.
```

Listing 3.1: Example HTML & JavaScript question text with variables

3 Development of a Quiz

Speed, Distance and Time

The screenshot shows a quiz question interface. At the top right, there are two buttons: 'Deactivate' and 'Edit Question'. The question title 'Speed, Distance and Time' is on the left, and '2 pts' is on the right. The question text asks for the speed of a car traveling 8 km in 3 hours. It lists the speeds of three students: John (13 km/h), Patrick (15 km/h), and Dany (9 km/h). Below the question, there are two input fields: one for the car's speed and one for the average speed of the students, with a 'Submit' button next to the second field. The 'Solution' section follows, showing the formula for velocity and the calculation for the average speed. At the bottom left, there is an 'Active' status indicator, and at the bottom right, there are 'Deactivate' and 'Edit Question' buttons.

Speed, Distance and Time 2 pts

How fast is a car driving (in km/h) when it travels 8 km in 3 hours?

These students move at different speeds:

- John moves at the speed of 13 km/h
- Patrick moves at the speed of 15 km/h
- Dany moves at the speed of 9 km/h

Calculate the average speed of the group.

The car drives with km/h

The average speed of the students is km/h

Solution

The solution for the car is calculated with the formula:

$$velocity = \frac{distance}{time} = \frac{8}{3} = 2.6666666666666665$$

The solution for the students is calculated with the formula:

$$\frac{13 + 15 + 9}{3} = \frac{37}{3} = 12.333333333333334$$

Active Deactivate Edit Question

Figure 3.7: Question view for an example question

The function must return an object with member variables named the same as the variables used in the quiz text. See an example for Q in listing 3.2. When the question is saved, the user is redirected to another page, where the function is executed along with other functions described in the forthcoming section. In this process the "Number of Preview Parameters" becomes relevant. This input field is located above the input area for the question parameter function in the quiz form (depicted in 3.6) and specifies how many times the parameter function should be executed. In every iteration, the object that is produced and returned by the parameter function is stored in a list and after all iterations, this list is posted to the question controller, where all parameters are stored in the database as "Preview Parameters".

3 Development of a Quiz

```
1 var Q = {  
2   carDistance : Math.floor(1 + Math.random() * Math.floor(15)),  
3   carTime : Math.floor(1 + Math.random() * Math.floor(3)),  
4   students : ['John', 'Patrick', 'Dany'],  
5   studentSpeeds: [13, 15, 9]  
6 }  
7  
8 return Q;
```

Listing 3.2: Example Q Function

The preview parameters are displayed on the question view right below the question and are intended to reveal errors in the parameter and solution generator functions. These values will not be used in quizzes as the quizzes generate their own parameter and solution set. The preview parameters serve solely for preview purposes. They are rendered beneath the question as shown in figure 3.8.

What is also visible in the preview, is the output of the solution function $S(Q)$, which will be described in the subsequent section.

The Question Solution Function $S(Q)$

The idea behind the solution function is that for every generated question parameter instance, there also exists a solution to the question for that specific question parameter instance. The described solution is generated when the parameter itself is created. The solution function takes the produced parameter Q as input and based on the values of Q , calculates the correct solution to the problem asked in the question text. Once the solution is calculated, it is stored in the database with a reference to the question parameter instance. If the functions Q and $S(Q)$ were set up properly, there exists one question solution for every question parameter that was generated for the question.

An example solution function is shown in listing 3.3. In this example, an object S is created which involves the computed solution for the speed of the car, which was asked in the example question above. Every member variable in the S object is considered as the correct solution to the problem

3 Development of a Quiz

Question Parameters Preview (Q)

See the list of generated parameters from the parameter function below. These are preview only parameters to verify your Q, S and G functions.

In an quiz scenario there will be a separate option to generate parameters.

| ID | Value Q | Solution S(Q) | Created Time |
|------|--|--|---------------------|
| 2008 | <pre>{ "carDistance": "9", "carTime": "1", "studentSpeeds": ["13", "15", "9"], "students": [</pre> | <pre>{ "carSpeed": "9", "groupSpeedAvg": "12.33333333" }</pre> | 2019-07-29 03:41:58 |
| 2009 | <pre>{ "carDistance": "7", "carTime": "2", "studentSpeeds": ["13", "15", "9"], "students": [</pre> | <pre>{ "carSpeed": "3.5", "groupSpeedAvg": "12.33333333" }</pre> | 2019-07-29 03:41:58 |

Figure 3.8: Question preview parameters with their solutions below the question text in the question view

asked in the question and the object must be returned at the end of the input field.

In the solution function, as in the grading function (that will be described later in section 3.2.3, contrary to the question text, no variable placeholders are used. Instead, it consists of pure JavaScript code and the object Q will be provided by the quiz application. Once the question is saved, the user is redirected to the next page (see figure 3.9), where all question parameter instances and subsequently their solution instances, are created and stored in the database.

This page is needed to perform the iterations of the question parameter and question solution process. The generation happens with JavaScript and includes the custom code entered by the user. All errors that are catchable are displayed on this page. Other propable errors can be viewed in the JavaScript console of the executing web browser.

3 Development of a Quiz

```
1
2 var S = {
3     carSpeed: Q.carDistance / Q.carTime ,
4     groupSpeedAvg: calculateGroupSpeed(Q.studentSpeeds) ,
5 }
6
7 function calculateGroupSpeed(speedArray) {
8
9     var speedSum = 0;
10
11     for(i in speedArray) {
12         speedSum += speedArray[i];
13     }
14     return speedSum / speedArray.length;
15 }
16
17 return S;
```

Listing 3.3: Example S(Q) function

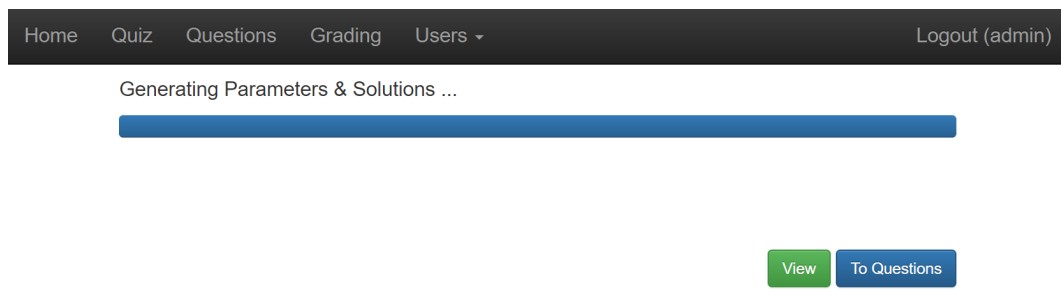


Figure 3.9: Question preview parameters and solutions are generated on a dedicated page

The User Input Area U

It is necessary to provide the possibility for the users to enter their answers to the question. These inputs can be made with the use of a customizable HTML input form that is appended below the question text. The two supported HTML input types are "hidden" and "text". This is relevant on the page for quiz attempts. In the case that a quiz is resumed and the user has previously answered questions, the given answers are loaded into the corresponding input fields. The type of an HTML input field determines

3 Development of a Quiz

```
1 The car drives with <input name="carSpeed" value="" /> km/h  
2 <br/><br/>  
3 The average speed of the students is <input name="groupSpeedAvg"  
   value="" /> km/h
```

Listing 3.4: Example user input area U

how it can be populated with a value. A checkbox field cannot be filled by applying the same logic as used for populating a text type input field. This implies that the type of the input field must be known to provide loading of previous answers for input fields of different types and no mechanism was implemented to support this behaviour. If other forms of inputs are used, the creator of the question must provide a way to map these inputs in the supported field types. The content of the user input area field in the question form is rendered into an HTML form element. The form element has one submit button which serializes all input fields contained in the form and posts it to the question controller which stores the user answers in the database.

The example shown in 3.4 contains two input elements, written in HTML. No question variables are supported in the user input area, but JavaScript code with custom logic to populate the input elements can be applied by the creator of the question. The names of the input elements are very important, as they are used to compare the given user answers with the correct solutions in the grading function described in the following section.

The user inputs are problematic in multiple ways. It is for example possible to resume an aborted quiz try and in this case, it could be irritating for the users, if they do not see their previously entered inputs when resuming the quiz. This requires the application to take the formerly entered inputs and load them into the corresponding input field, when the quiz is loaded in the resume state, which in turn implies that the quiz application has to know the type of input as it cannot prepopulate the input fields the same way when checkboxes, radio buttons or other types of inputs were used compared to inputs of the type text or hidden. In any case, the quiz application attaches a marker to a question that has been answered previously.

One requirement was that there should be a possibility to include custom

3 Development of a Quiz

logic that is executed at the time when a question is submitted in a quiz. To enable this, an additional input field was created in the question form. It is called "User Answer Submit Hook H" and can be populated with JavaScript code. The entered code is then executed when users submit the question.

The Grading Function $G(U,S)$

This member of the content generators is called the "Grading Function $G(U,S)$ ". It is necessary to automatically calculate grades for user answers to a question and compares the given user input U to the correct solution S .

Listing 3.5 shows an example implementation of the grading function for the example used for all of the content generator elements. The function must be written in JavaScript and return a floating-point value between one and zero.

```
1 var grade = 0.0;
2
3 if (Math.abs(U.carSpeed - S.carSpeed) <= 0.1 ) {
4     grade += 0.5;
5 }
6
7 if (Math.abs(U.groupSpeedAvg - S.groupSpeedAvg) <= 0.1 ) {
8     grade += 0.5;
9 }
10
11 return grade;
```

Listing 3.5: Example grading function $G(U,S)$

The standard outcome service¹ of the LTI protocol only accepts floating-point values ranging from 1 to 0, resembling 100% to 0% and therefore the quiz application also stores all grades in this floating-point format. This explains why the grading function must return a floating-point value within that range. With this function, the assessment of the user answer is completely in the hands of the question creator and the application itself has no other grading mechanism implemented for questions.

¹LTI Outcome Service Specifications <https://www.imsglobal.org/specs/ltiomv1p0/specification> 01.07.2019

3 Development of a Quiz

The Solution Guide SG(Q)

The solution guide is the last content generator of the question form which is described in this section. Its purpose is to show the users the correct solution to a question. If the teacher wishes to do so, he must define an HTML solution that is rendered for the users when they visit the solutions page after a closed quiz try, if the quiz option "Show Solutions" was activated.

```
1 <script>
2   var velocity = {{carDistance}} / {{carTime}};
3   var mexp = "velocity = \\frac{distance}{time} = \\frac{{{
4     carDistance}}}{{{carTime}}} =" + velocity;
5
6   var studentFrac = "\\frac{";
7   var studentSpeedSum = 0;
8   var studentSpeeds = {{studentSpeeds}};
9
10  for(i in studentSpeeds) {
11    if(i > 0) {
12      studentFrac += "+";
13    }
14    studentFrac += studentSpeeds[i];
15    studentSpeedSum += parseInt(studentSpeeds[i]);
16  }
17  studentFrac += "}{} = \\frac{" + studentSpeedSum + "}{} = "
18  + studentSpeedSum / 3;
19 </script>
20
21 The solution for the car is calculated with the formula:
22 <script>
23   document.write('$' + mexp + '$');
24 </script>
25
26 The solution for the students is calculated with the formula:
27 <script>
28   document.write('$' + studentFrac + '$');
```

Listing 3.6: Example solution guide

The question view, quiz attempt and quiz solutions pages include the

3 Development of a Quiz

MathJax² library, which can be used to pretty-print mathematical formulas. The use of the MathJax notation is demonstrated in listing 3.6 and results in the output shown on the panel below the question (see figure 3.10). The example solution guide also contains question variables that are replaced by the corresponding question parameter values when the solution guide is rendered.

Solution

The solution for the car is calculated with the formula:

$$velocity = \frac{distance}{time} = \frac{9}{1} = 9$$

The solution for the students is calculated with the formula:

$$\frac{13 + 15 + 9}{3} = \frac{37}{3} = 12.333333333333334$$

Figure 3.10: Question solution guide for the example question

Points and Estimated Time

Closing this section there are two fields that have not been described yet: The question points and time estimate. These values are aggregated on the top of the quiz view page, to gain insight on how long the quiz would approximately take (estimated time) and how difficult would be (total points). The values only serve as an orientation for the quiz creators and have no other technical purpose.

Activation of Questions

After all of the information is entered for the question and the content generators are set up correctly, the question is ready to be activated. Activation means that the question will be visible in the quiz question selection

²MathJax Homepage <https://www.mathjax.org> 01.07.2019

3 Development of a Quiz

and thus can be used in quizzes. To prevent unwanted side effects from deactivation, questions that are already assigned to quizzes will not disappear from the affected quiz, if the question is deactivated. It just cannot be reassigned to further quizzes. After a question is activated, a success message is displayed to the users and the question setup is considered to be complete.

Editing Questions and Testing

Once questions have been created, every part of them can be edited. The question edit form is identical to the new question form and every time the question is saved, the existing question parameters (produced by the question parameter function described in 3.2.3) are deleted and new ones created because it is hard to detect changes in the generator functions for parameters and solutions. Thus the decision was made that the question parameters and their solutions are generated every time the question is edited and saved. This has no direct effect on the quiz parameters described in section 3.2.4, to avoid the invalidation of quiz results.

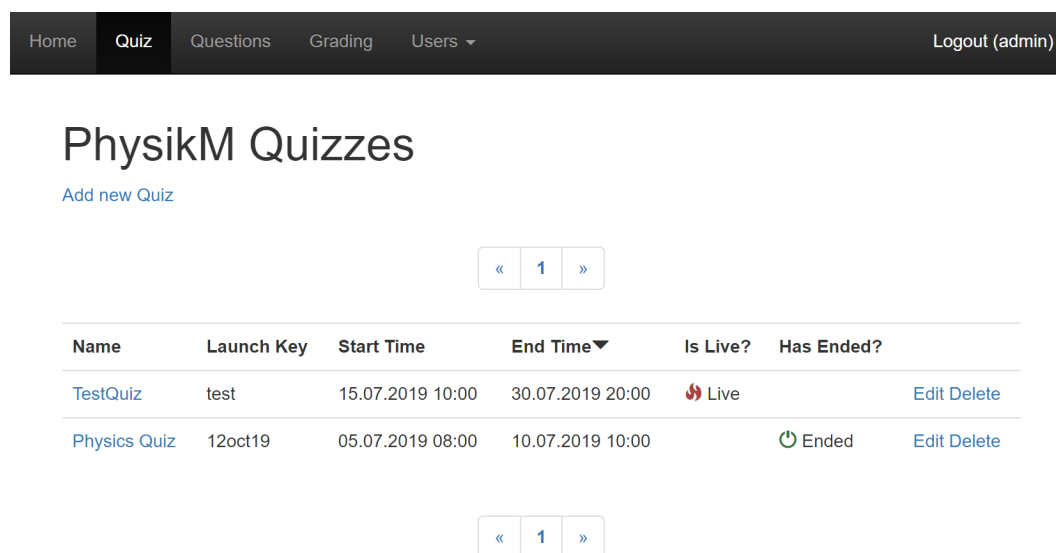
Testing the questions is important to avoid the invalidation of quiz results and malfunctioning of quizzes. Instructors should always keep in mind that, as they gain flexibility with programmable questions, they also gain room for errors. Therefore in the question (and finally again in the quiz preview), they have the possibility to test user inputs and see the outcomes of their programmed functions. When they enter their testing answers into the input fields of the question and submit these, a preview modal window is rendered. It contains information about what the application received as a user answer in form of JSON, what the solution looks like in JSON, what the output of the grading function for that particular example and how many points result from that grade would be. The received points are simply calculated by multiplying the floating-point grade value (ranging from 0.0 to 1.0) by the points defined for the question.

3 Development of a Quiz

3.2.4 Creating Quizzes

Quizzes can be created by instructors and administrators and resemble an exam scenario, consisting of just one, or multiple questions. Once set up, a quiz can be published and be launched by an LTI launch request directed towards the quiz application with a set custom parameter that identifies the quiz. The quiz creation process is divided into four steps which are described in this section.

To create a quiz, an instructor or admin can navigate to the quiz index page by clicking the top menu item "Quiz", which will show a paginated list with current quizzes, sorted by their end time in descending order (see figure 3.11).



The screenshot shows a web interface for 'PhysikM Quizzes'. At the top is a navigation bar with 'Home', 'Quiz', 'Questions', 'Grading', and 'Users' (with a dropdown arrow), and a 'Logout (admin)' link on the right. Below the navigation bar is the title 'PhysikM Quizzes' and a link 'Add new Quiz'. A pagination control shows '« 1 »'. The main content is a table with the following data:

| Name | Launch Key | Start Time | End Time | Is Live? | Has Ended? |
|--------------|------------|------------------|------------------|----------|---|
| TestQuiz | test | 15.07.2019 10:00 | 30.07.2019 20:00 | 🔥 Live | Edit Delete |
| Physics Quiz | 12oct19 | 05.07.2019 08:00 | 10.07.2019 10:00 | 🕒 Ended | Edit Delete |

Below the table is another pagination control showing '« 1 »'.

Figure 3.11: The list of quizzes with indicators for live and ended quizzes

Quizzes can have multiple states and the list of quizzes shows their most important information. A quiz is considered "Live" if it is published and the current time lies between the start and end time of the quiz. In this state, any editor should be extremely careful in regard to changing the quiz, as a student could potentially be doing that quiz at the moment and when answers were already given, the grading of the quiz could lose its fairness.

3 Development of a Quiz

The problems of editing quizzes that are live are described in more detail in section 3.2.4.

Above the list of questions, located on the left side, there is a link with the label "Add new Quiz", which leads to the form for creating a new quiz as shown in figure 3.12.

The screenshot shows a navigation bar with 'Home', 'Quiz', 'Questions', 'Grading', and 'Users' (with a dropdown arrow). Below the navigation bar is the title 'Add new Quiz'. The form contains the following fields:

- Quiz Name:** Input field containing 'Final Quiz'.
- Launch Key:** Input field containing 'midTerm2019'.
- Start Time:** Input field containing '01.06.2019 13:00'.
- End Time:** Input field containing '01.06.2019 15:00'.
- Tries:** Input field containing '1'.
- Show Solutions after Try?:** A checkbox that is checked.
- Add:** A blue button to submit the form.

Figure 3.12: Creating a new quiz

This page is the first step of producing a quiz and only basic information is added. Next to the quiz name, which is displayed when students try the quiz, there is an input field for the very important launch key. This key identifies the quiz in an LTI launch scenario and must be set up on the LTI consumer side as well. If the launch key of a quiz changes and is not

3 Development of a Quiz

adapted on the consumer side, the quiz cannot be launched anymore from the connected application.

Quiz Participation Period

The two fields "Start Time" and "End Time" are important as well, as they define the quiz participation period. A quiz can only be started when the current time is greater than the start time and smaller than the end time. The quiz application only counts in minutes, so seconds are spared in the quiz configuration. Start and end time must be entered precisely in the format "d.m.Y H:i". This notation relies on the PHP date format³. That means for example "26.06.2019 13:00" is a valid date and "26.06.2019 13:00:01" is considered as an invalid date by the application.

When comparing against the starting time, the application identifies the quiz as started at the exact minute of the starting time. A quiz with the start time of 13:00 o'clock, can be launched at 13:00:00 o'clock but not at 12:59:59 o'clock.

The end time, on the other hand, is compared in a more forgiving way. When the full minute of the end time is over, the quiz cannot be launched, no answers can be submitted and the quiz try cannot be closed anymore. This means that a quiz with the end time of 14:30 can be launched or finished until 14:30:59, but not at 14:31:00 o'clock.

Number of Retries and Show Solutions Option

The two remaining input fields on the quiz creation page are the number of "Tries" and the "Show Solutions after Try" checkboxes. The number of tries specifies how many tries a user has before the quiz becomes locked for that user. If the number of tries is set to one, every user has only one try for this quiz, if the number is set to zero, no-one can start that quiz. A quiz with just one try is considered the type of quiz that is closest to a classical, summative examination scenario with a formal assessment by a teacher or

³PHP Date Format <https://www.php.net/manual/en/function.date.php> 01.07.2019

3 Development of a Quiz

tutor. If retries are permitted, the quiz has properties of a self-assessment scenario as per the description in section 2.2 on retries.

When all fields are filled out and the quiz is successfully saved, the user is redirected to the view quiz page, which serves as a preview and testing tool for the quiz. At first, it looks empty because there are no questions assigned to the quiz yet, but when questions are assigned, it looks like depicted in figure 3.13.

The screenshot shows a web interface for a quiz. At the top, there is a navigation bar with 'Quiz', 'Questions', 'Grading', and 'Users' menus, and a 'Logout' link. Below the navigation bar, there is a breadcrumb trail: 'Home / Quizzes / View Quiz 20'. The main heading is 'Preview Quiz: Physics Quiz'. To the right of the heading are two buttons: 'Unpublish' and 'Edit Quiz'. Below the heading, there are two columns of quiz parameters. The left column contains: 'Total Points: 4pts', 'Total Time Estimate: 7m', and 'Tries: 10'. The right column contains: 'Start Time: 05.07.2019 08:00', 'End Time: 30.07.2019 10:00', and 'Launch Key: 12oct19'. Below these parameters, there are three status indicators: 'Published' (green), 'Showing Solutions' (blue), and 'THIS QUIZ IS LIVE' (red with a warning icon). The main content area is titled 'Speed, Distance and Time' and is worth '2 pts'. The question text is: 'How fast is a car driving (in km/h) when it travels 15 km in 2 hours?'. Below the question, it says 'These students move at different speeds:' followed by a bulleted list: 'John moves at the speed of 13 km/h', 'Patrick moves at the speed of 15 km/h', and 'Dany moves at the speed of 9 km/h'. The question asks to 'Calculate the average speed of the group.' There are two input fields: 'The car drives with [] km/h' and 'The average speed of the students is [] km/h'. A 'Submit' button is located to the right of the second input field. At the bottom left, there is a link 'Question Parameters Preview (Q)' and a plus sign icon.

Figure 3.13: The quiz preview page, showing a live quiz

3 Development of a Quiz

Assigning Questions

To navigate to the question assignment page the user has to click either on the edit link in the quiz overview or on the edit button on the top or the bottom of the quiz preview page. The edit page (figure 3.14) shows a list of active questions and an initially empty list of selected questions. The lists are made with the help of jQueryUI Sortable ⁴ elements and the list elements can be moved between the active question pool and the selection by dragging them from one and dropping them on the other. In case the question pool grows and the list of active questions becomes confusing due to too many entries, there is a possibility to filter the question selection by entering a search term above the active question list. When entering more than two characters into the filter bar, a live search is triggered that searches for matching question titles, limiting the possible selections.

Already assigned questions are excluded from the available active question list, to prevent duplicate assignment of the same question in one quiz. Once all desired questions are assigned, the user must click on the "Save Changes and continue to Parameter Quantity" button. This leads to the final step of the quiz creation before it can be published promptly.

Generating Quiz Parameters, Testing and Publishing

In the last step of quiz creation, the user must specify how large the set of random variables should be for each question of the quiz. The form is placed here because at this point it is most likely that the teacher already knows how many students will participate in the quiz, which might not be the case at the time of question creation. The parameters for each question of the whole quiz are called the quiz parameter set and the size of the set is defined through the number of quiz parameters for each question. The reason for this new type of parameter, even though there already are question parameters that were created earlier, is to decouple the quiz and subsequent quiz attempts of users, from the questions. This way, when questions are edited, they do not compromise existing quiz attempts and user answers,

⁴jQueryUI Connected Sortable <https://jqueryui.com/sortable/#connect-lists>
01.07.2019

3 Development of a Quiz

Edit Quiz

[Save Changes and continue to Parameter Quantity](#)

Quiz Name
Physics Quiz

Launch Key
12oct19

Start Time
05.07.2019 08:00

End Time
30.07.2019 10:00

Tries
3

Show Solutions after Try?

Active Questions
Filter:

| | | |
|------------------------------|----|-----|
| Feynman Diagrams | 5p | 10m |
| Kinetic and Potential Energy | 5p | 10m |
| Electromagnetism | 5p | 15m |
| Input Elements | 1p | 2m |

Selected Questions

| | | |
|--------------------------|----|----|
| Speed, Distance and Time | 2p | 5m |
| The Force of Gravity | 2p | 2m |

[Save Changes and continue to Parameter Quantity](#)

Figure 3.14: The question assignment page

which is necessary for persistently storing the quiz results. When attempting a quiz, the learner gets one random quiz parameter for every question and the selection is remembered to verify the results later, based on the random variables that were shown to the student at the point of taking the quiz.

After this last step, the quiz parameters are generated by calling each question's content generator functions Q and $S(Q)$. All catchable JavaScript exceptions that occur in this process are printed on the screen. This unfortunately, does not cover syntax errors, which are very common. At this stage, the questions should already be error-free because the testing should have been done in the question preview. It is the duty of the instructor who creates and reviews the quiz to validate that every question works as intended in the quiz environment. To help the instructors with the task of validating the correctness of the questions in the quiz environment, it is

3 Development of a Quiz

possible to submit answers in the quiz preview. The submission triggers a modal window (depicted in 3.16), which contains information about the submitted answer, the correct solution, the calculated grade and the received points for that answer. Now that the quiz parameters are generated and the quiz was tested, the user can publish the quiz. This means that if the quiz participation period has started, students can try the quiz. The publishing action has no effects other than preventing or enabling the start process of a quiz and once a quiz was published or unpublished, a success message is displayed to the user who triggered the process.

The Force of Gravity 2 pts

How strong is the force of gravity for an object with the mass 31 kg?

N

Question Parameters Preview (Q) +

See the list of generated parameters from the parameter function below. These are preview only parameters to verify your Q, S and G functions.
In an quiz scenario there will be a separate option to generate parameters.

| ID | Value Q | Solution S(Q) | Created Time |
|------|--------------------------------|-------------------------------------|---------------------|
| 2060 | <pre>{ "mass": "31" }</pre> | <pre>{ "force": "304.11" }</pre> | 2019-07-29 04:27:15 |
| 2061 | <pre>{ "mass": "136" }</pre> | <pre>{ "force": "1334.16" }</pre> | 2019-07-29 04:27:15 |

Figure 3.15: Collapsible list of generated quiz parameters

3 Development of a Quiz

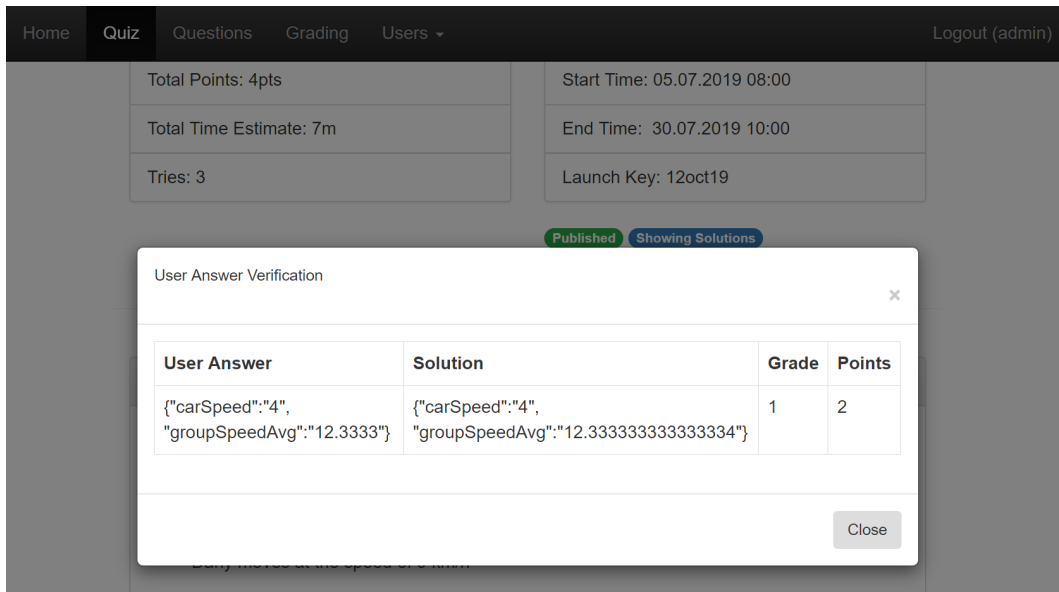


Figure 3.16: Example input made on the quiz preview page to test the outcome of the answer

3.2.5 Taking a Quiz

Quizzes can be started by calling the general LTI start URL (described in 3.2.1) for the application with an LTI custom parameter. This custom parameter, the so-called "launch key", is used to identify the quiz that should be started and is defined on one end, in the quiz creation process and on the other end, in the LTI consumer.

If the quiz with the given launch key cannot be found, an error message is displayed. If the quiz is found, some checks are performed to make sure that the user has the right to start this quiz. These checks include:

- Check if the quiz has been published
- Check if the quiz start time lies in the past or is now (by the minute)
- Check if the quiz end time lies in the future or is now (by the minute)
- Check if the user either has enough tries left or currently has an active try

3 Development of a Quiz

If all of the above checks are passed, the user sees the start page for quizzes as shown in (figure 3.17). If one or more checks fail, the user sees an according, user-friendly error message, that explains why the quiz cannot be started. Common reasons for this could be that the participation period has ended, or the user has already reached the maximum number of tries for that particular quiz.

© 2019 - 2019 Graz University of Technology

Figure 3.17: The quiz start page in the resume state

At this point, the user has some options depending on the current quiz state and the settings that were made at the time of quiz creation. If the display of solutions was allowed and there is at least one closed quiz try, the "Show Solutions" panel is displayed. This enables the learners to self-assess their quiz performance and does not require teacher interaction, under the assumption that the quiz was set up properly by the teachers prior to the start of the quiz.

The combination of multiple retries and well-prepared solutions could lead to an overall increase of final exam pass rates in a self-assessment scenario (Ćukušić, Garača, and Jadrić, 2014). The quality of information (Sun et al., 2008) and the system that delivers the information (Alsabawy, Cater-Steel,

3 Development of a Quiz

and Soar, 2016) are affecting the perceived usefulness of e-Learning systems and the e-Learning experience, thus this part of the application was built with a focus on intuitiveness, visual appeal and responsiveness to increase the perceived system quality. The information quality is in this case, largely based on the teacher who creates the questions and their solutions.

On the start quiz page as shown in figure 3.17, the quiz state is in resume mode. The two other states that influence the labeling and color of the buttons are the "Initial Try" and "Retry" state. To increase the perceived intuitiveness, all buttons with no click consequence are colored in green. Buttons with click consequences are colored in orange (see figure 3.18) and if severe consequence is to be expected, the button is colored red. The blue primary button color that is used, for example, on the first quiz attempt creates a quiz try, but the process can be considered as the normal workflow and unobjectionable.

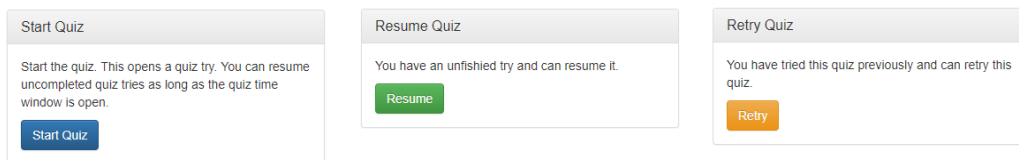


Figure 3.18: Severity levels of buttons

Figure 3.19 depicts the quiz flow.

The quiz attempt starts with the page shown in figure 3.17). When the user starts the quiz by clicking the according button, he or she is redirected to the next step which is labelled as the "Quiz Attempt" in figure 3.19.

A few steps are accomplished in the instant the quiz attempt is started:

- It is verified that the quiz, referenced in the LTI launch request, is the started one
- Either a new quiz try is generated, or an active quiz try is loaded
- The quiz is loaded with all questions and random quiz parameters
- In case of an active quiz try, the previously given user answers are populated into the corresponding input fields of the questions
- An indicator is placed at the already answered questions to signalize that the question was answered previously

3 Development of a Quiz

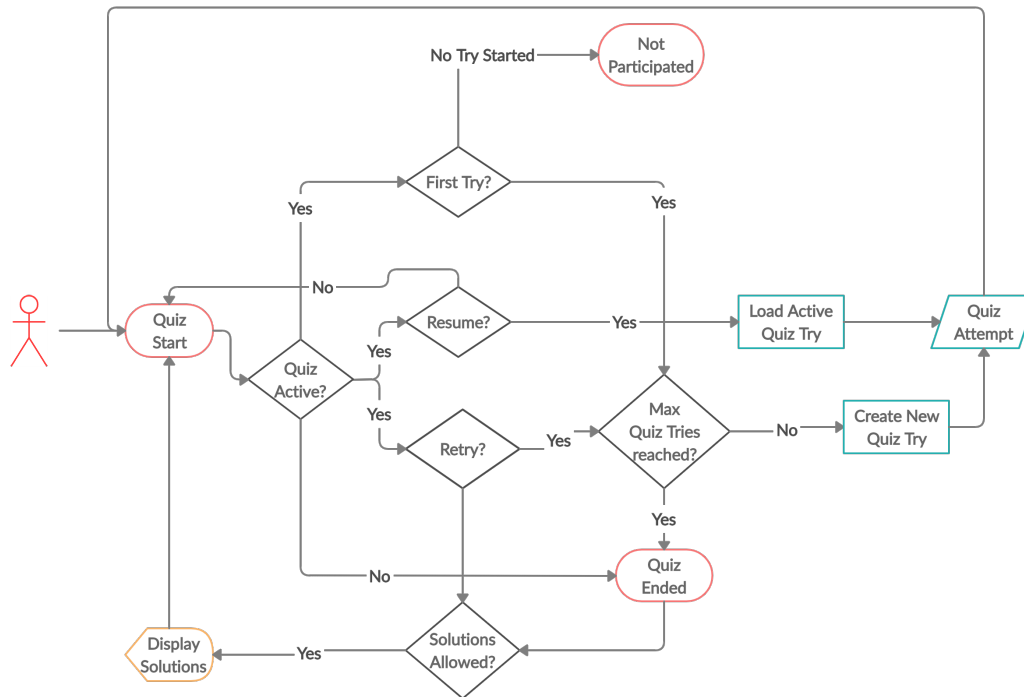


Figure 3.19: The quiz flow

The user that attempts the quiz can now see all of the questions with the variable placeholders replaced by values that were randomly selected from all variables for the question in the quiz parameter set. The random selection is stored along with the current quiz try so that the learner always has the same values for a particular attempt, even if he or she closes the browser window, loses internet connection or faces similar problems. Only when the quiz try is ended by clicking the "End Quiz" button and the choice is confirmed, it is possible to get new random values in a retry scenario. This behavior is visualized in figure 3.19 by the elements "Load Active Quiz Try" and "Create New Quiz Try".

Every question has an HTML form prepared in the bottom section, which contains all input fields that were added to the user input area in the question creation process and one submit button. When pressing this button the user submits all input fields of that question and dispatches them with

3 Development of a Quiz

an AJAX request to the corresponding controller, which stores the answers in the database in JSON format. When answers are submitted, the result of the submit action is printed in a modal window that pops up when the request is completed.

After all questions are answered or the learners decide to end the quiz for other reasons, they are again redirected to the quiz start page, that depending on the state of the quiz, might now display the solutions to the last closed quiz try and look slightly different than before. If no quiz tries are left, an error message will be displayed that explains that the quiz cannot be started anymore and the quiz is over.

3.2.6 Showing Solutions

At the point where students have completed a quiz try, they can see the correct solutions to the problems that were part of the quiz, if the creator of that quiz has allowed to display the solutions. The solution page contains the student's answers along with the correct solution provided by the teacher at the question setup and how many points the student has received for the provided answer. The points that the students see are calculated by the grade that is returned by the grading function of the question multiplied by the question points. For example, in case the student made a partially correct answer and received a grade of 0.5 by the grading function and the question is worth 2 points, the student will receive 1 point.

The solution page shows the name of the quiz on the top (see figure 3.20) followed by every question of the quiz and within the question container, all the information about the correct solution, student answer and received points.

3.2.7 Processing the Quiz Results

In the former sections of this chapter, the creation of the learning material and the quiz attempt were described. The whole process leads to the point where a quiz is closed and a teacher can analyze the results of the students.

3 Development of a Quiz

Solutions for the Quiz: Physics Quiz Back to Quiz Start

#1 Speed, Distance and Time 2 pts

How fast is a car driving (in km/h) when it travels 4 km in 3 hours?

These students move at different speeds:

- John moves at the speed of 13 km/h
- Patrick moves at the speed of 15 km/h
- Dany moves at the speed of 9 km/h

Calculate the average speed of the group.

Correct Solution:

The solution for the car is calculated with the formula:

$$velocity = \frac{distance}{time} = \frac{4}{3} = 1.3333333333333333$$

The solution for the students is calculated with the formula:

$$\frac{13 + 15 + 9}{3} = \frac{37}{3} = 12.333333333333334$$

Your Answer: Received Points:

carSpeed: 0.75 1

groupSpeedAvg: 12.3333

Figure 3.20: The solution page

The quiz application is designed to produce grades for single answers to questions, as well as the overall quiz. This means that there are two types of grades that will be called "Question Grades" and "Quiz Grades" in this section.

At the time that a quiz has ended, instructors or administrators can navigate to the top menu item "Grading" where they will find a list of quizzes. The list looks very similar to the quiz overview list shown in 3.11 and indicates if a quiz has ended or is live. From there, they can select the quiz they want to grade, by clicking on its name. This will lead to the quiz grading page for that quiz (figure 3.21). The page explains how to grade a quiz and submit

3 Development of a Quiz

the results to the connected LTI consumer.

Quiz Questions **Grading** Logout

Grade Quiz: Physics Quiz

Step 1:
Produces automatically calculated grades that originate from the grading function G defined in the according question. Resulting values should be within a range from 0.0000 - 1.0000

Step 2:
Review all calculated grades and override them with a value of your choice if needed (again between 0.0000 and 1.0000). Click the save button next to the grade you want to change to store it.

Step 3:
Review the final grades for the quiz. This is a single grade for every student that participated. You can override the total quiz grade with a value between 0.0000 and 1.000 if needed.

(1) Grade Question Answers
Produces the grades for all user answers given at this quiz. They are generated by the grading function that was set up for the question. You can override the grades manually.
Grade Questions

(2) Review Single Answer Grades
The grades for the quiz are the average over the single question grades and are calculated automatically. You can override them manually.
Review Question Grades

(3) Review & Publish Quiz Grades
Make sure that all final grades are correct, override them if needed and publish them to the Tech Center.
Review & Publish Quiz Grades

Figure 3.21: The quiz grading page

The grading process is again divided into separate steps.

- Calculation of all question grades
- Manual review of the calculated Questions
- Manual review of the quiz grades and publishing

Step 1: Calculation of Question Grades

The grading process starts by navigating to the first page of the grading process. On this page, all user answers are loaded and all JavaScript grading functions of all questions are inserted into the page. The grades are then calculated by these functions comparing the users' answers with the correct

3 Development of a Quiz

solutions that were calculated for the quiz parameter that was assigned to the user on the quiz take. The results of the calculation are stored in the database and assigned to a particular quiz try. In this stage, grades are calculated for every single quiz try, regardless of its properties. This is done with the thought, that in the future, different modes for quizzes could be implemented. Only the latest closed quiz try will be of relevance in the next step: the manual verification of grades.

Step 2: Verification and Correction of Question Grades

After all question grades were calculated in step one for all tries and all answers by users, in the next step a teacher can review the automatically calculated grades and override them if necessary. The interface shows the answers of the participants of the quiz, grouped by questions, in a collapsible table. Each row in the table represents a student's answer to a question. The single values in the JSON structures are sorted by name to ease the process of comparing them. Each question grade can be overridden manually and if the grade was overridden once, any recalculation of the automatic grade will not delete the manual grade. To remove it, the value must be manually deleted from the input field and after that, be saved again.

Step 3: Quiz Grades and Publishing

The last step of a complete use case of the application is manifested in the final quiz grade page. The page displays the final quiz grades, which are calculated as the average of the grades over the number of questions in a quiz. The resulting grade is a value between zero and one, indicating the percentage of the learner's success. Teachers have the possibility, just like at the question grades, to overwrite the final grades. Here the same mechanism for manual grades as for questions grades apply. Manual values are not deleted, and the automatic values can be always refreshed without endangering the manual grade. Each row in the list resembles the final quiz result of one participant. With the "Publish" button, the grades are sent to the LTI consumer that started the quiz. As a grade replace request is used to submit the quiz grades, the grades can be resubmitted as often as

3 Development of a Quiz

Review Question Grades

Physics Quiz

Please review the automatically calculated grades and override them if necessary. To override a grade use the "Override Grade" column of the grade entry you want to change. Grades must be a value between 0.0000 and 1.0000. The final quiz grade will be the average grade over the number of questions in the quiz

Every row you see represents the student answer of their last completed quiz try.

To Quiz Grades

| UserID | TryID | User Answer | Solution | Calculated Grade | Override Grade |
|--------|-------|---|---|------------------|---------------------------|
| 7 | 127 | <pre>{ "carSpeed": "13", "groupSpeedAvg": "12.3333" }</pre> | <pre>{ "carSpeed": "13", "groupSpeedAvg": "12.33333333333333" }</pre> | 1 | <input type="text"/> Save |
| 4 | 124 | <pre>{ "carSpeed": "0.75", "groupSpeedAvg": "12.3333" }</pre> | <pre>{ "carSpeed": "1.3333333333333333", "groupSpeedAvg": "12.33333333333333" }</pre> | 0.5 | <input type="text"/> Save |

Figure 3.22: The question grade review page, comparing students' answers with correct solutions

needed without any risk. The grades are overridden in the connected LTI consumer, which is identified by the LTI outcome service URL, transmitted in the launch request of the quiz.

3.2.8 Administering Users, Roles and Permissions

The concluding topic of this section is user management. Instructors and Learners do not have access to this menu item and only technicians should access it, in case there is a problem with the LTI connection of users, or the application is extended with new features that require additional setup regarding roles and permissions. The users menu contains three sub-items and allows an administrator to manage users, roles and permissions.

Changing entries in this category can potentially break the application and

3 Development of a Quiz

must be done with increased care, as it would even be possible to lock out an administrator from the application. The user management described in this section was taken from the free open source book (Oleg, 2018) and modified to the needs of the quiz application. Since not all features were needed, their implementation was omitted.

Users

The users section contains a paginated list of all users that were created in the application and the possibility to manage them. There are two types of users: LTI Users and Application Users. Application users are rare and intended only for setup and support purposes of the application. For example in the case that the LTI communication breaks down for any reason, there still is the possibility for an administrator to log into the application and perform administrative tasks. On each LTI connection, for example, a Quiz launch, a new user is created if no user with that LTI user identification does exist yet. If a user already exists, the roles are updated correspondingly to the information that was contained in the LTI launch request. For example, if a connected user in the invoking LTI consumer is an Instructor, his or her user entry in the quiz application will be mapped to the instructor role of the quiz application. If the same user reconnects on another day, as a learner in the LTI consumer, he will be associated only to the learner role and the reference to the instructor role is deleted. Multiple role assignments to one user are possible.

Roles and Permissions

Roles in the quiz application are assigned to users and one user can have multiple roles. An inheritance mechanism enables a role hierarchy. The default setup for the hierarchy is as follows: The learners have only the permissions that are assigned to them directly, Instructors inherit all permissions from learners and administrators inherit all permissions from instructors. Every role is designed to have an LTI counterpart which is described by the field "LTI Name". When an LTI launch request reaches the application, after it is validated, the quiz application tries to find all

3 Development of a Quiz

roles that are corresponding to the roles element of the launch request and creates an assignment to the application roles.

The screenshot shows a web interface for managing roles. At the top, there is a dark header with 'Users' and a dropdown arrow. Below it is a breadcrumb trail: 'Home / Manage Roles / View Role Admin'. The main heading is 'View Role Admin', followed by an 'Edit Role' button. A table displays the role's details:

| | |
|---------------|---------------------------------------|
| ID: | 1 |
| Name: | Admin |
| LTI Name: | urn:lti:sysrole:ims/lis/Administrator |
| Description: | Must be able to perform every task |
| Date Created: | 2019-05-20 16:44:40 |

Below the table is the 'Permissions' section with an 'Edit Permissions' button. A table lists the permissions:

| Name | Description |
|---|---|
| <input checked="" type="checkbox"/> application.manage | Grants full access to Users, Roles & Permissions |
| <input checked="" type="checkbox"/> grade.manage (inherited) | Enables the possessor to edit and submit grades |
| <input checked="" type="checkbox"/> question.manage (inherited) | Can list, add, edit and delete questions |
| <input checked="" type="checkbox"/> quiz.manage (inherited) | Can list, add, edit and delete Quizzes |
| <input checked="" type="checkbox"/> quiz.take (inherited) | Allows the permission holder to take part in a quiz |

At the bottom, there is a copyright notice: '© 2019 - 2019 Graz University of Technology'.

Figure 3.23: Administrator role, showing inherited permissions and the LTI name

The quiz application has a built-in access filtering system, which is active at all times for every action. If any action violates the access rules, the user is redirected to an error page which states, that he or she does not have the permission to access this page. There should be no need to change anything on the permissions unless the application is extended with new features in the future. Removing or renaming existing permissions will cause malfunctioning of the access filter and due to its restrictive nature, blocks users from performing tasks related to the changed permission. The access filter compares if a user has the permission for the required action

3 Development of a Quiz

by comparing the name that is programmed into the application with the name of the permissions that the user currently has. This means that even the renaming of permissions can stop the filter from working.

3.3 Technical Perspective

This section elaborates on how the quiz application was made from a technical standpoint and describes the used technologies and the technically challenging points. The quiz application is a PHP application developed on and designed for Apache web servers, with a MariaDB or MySQL database instance installed.

3.3.1 Zend Framework

To create the quiz application described in chapter 3, Zend Framework 3 was used. Zend Framework 3 was released on the 28th of June 2016⁵ and is one of the most popular and long-lived PHP frameworks at this time. The open-source project was recently taken over by the Linux Foundation and at the time of writing this thesis, there is not much information available about the future of Zend Framework under the new leadership. The new project will be called "Laminas" and the goal is to get it operational in Q2 or Q3 of 2019⁶.

Zend Framework 3 uses an MVC structure and a modular approach to web development. The quiz application consists of three distinct modules, where one of the modules is the standard application module of the Zend Framework. The other two modules are called the "Quiz" and the "LTIApp" Module. The idea behind these modules is to create reusable pieces of code that can easily be ported to new applications.

The Zend Framework documentation itself provides a skeleton application that can be downloaded via Composer, a PHP package manager that is

⁵Zend Framework Blog <https://framework.zend.com/blog/2016-06-28-zend-framework-3.html> 03.07.2019

⁶The Laminas Project <https://getlaminas.org> 03.07.2019

3 Development of a Quiz

heavily utilized by the Zend Framework. The quiz application requires many common Zend Framework packages via Composer and one additional PHP package for the communication over the LTI protocol. The LTI framework that was used is the official LTI PHP framework⁷ developed by IMS Global, which invented the protocol.

The Quiz Module incorporates all models, views and controllers centered around the creation of questions, quizzes, question grades and quiz grades. The LTIApp module, on the other hand, contains all source code related to the LTI protocol and user management. The user management was mostly adopted from Oleg (2018), who created a comprehensive open-source book with examples about the usage of Zend Framework 3. The example was then modified in a way to support LTI authentication and login users into the quiz application based on the preliminary LTI launch request.

3.3.2 The IMS Global LTI Framework and User Authentication

To ease the work with the LTI protocol, IMS Global offers a PHP framework that can conveniently be added to PHP projects with the help of Composer. Once the framework was installed, the first step that should be done is the implementation of the framework class "ToolProvider". See the most basic implementation of this class in listing 3-7.

To successfully launch an LTI application, the first task is to validate the received request. This is done by comparing the OAuth signature of the request, with a generated signature of the received request, which is created using the consumer key and shared secret. If the signatures match, the request is valid (Vickers and Booth, 2014, p.6,7).

This is done by the framework if the implementation of the ToolProvider classes' **handleRequest()** method is called right after receiving the launch request. Following the guide provided by Vickers and Booth (2014) the next steps are to check the resource link that is identified with an ID in

⁷IMS Global LTI PHP Framework <https://github.com/IMSGlobal/LTI-Tool-Provider-Library-PHP> 03.07.2019

3 Development of a Quiz

the request and corresponds to the external tool activity in Moodle. The quiz application does not differentiate between resource links and therefore this check is omitted and all requests coming from the same consumer are categorized as one resource. The following recommended steps are more important for the quiz application and are performed as follows. If a user with the given LTI user ID is already present in the database, it is updated with new information from the launch request and the same user is used to create a session. If there are any currently active sessions, they are destroyed before a new one is created. In the case that there is no existing user in the database, a new record is created. Another point that is described in the implementation guide is that it should be verified that the user has access to the resource that is requested. This is handled by the built-in RBAC mechanism described in sections 3.2.8 and 3.2.8, that will block users from accessing actions for which they have no access rights. The final point that is recommended is to redirect the users to an appropriate entry page, which is done by leading students to the quiz page and administrators and instructors to the home page.

The behavior for launching the application which was described in this section is implemented in the **onLaunch** method of the custom implementation of the LTI ToolProvider base class. It is called automatically by the **handleRequest** method of the LTI framework's ToolProvider class.

The other functions that are displayed in listing 3.7 are the basic interface between any two LTI applications that are connected. The most important function is the formerly mentioned **onLaunch** function. The other functions are also called by the framework but on different occasions. For example, in the case of an LTI Registration Request, the **onRegister** method is called. This method should include all logic for handling a request to register a new LTI consumer or user and is used by version 2.0 of the LTI protocol. The **onError** method is called when an error happens within the execution of the **handleRequest** function. The implementation of the **onContentItem** function can be used for what the LTI documentation calls deep linking. This describes a behavior where external content is defined in the LTI consumer and then requested from an LTI tool provider. In this application, a similar behavior was achieved with the usage of an LTI custom parameter, which identifies the quiz that should be launched.

3 Development of a Quiz

```
1 class CustomToolProvider extends ToolProvider\ToolProvider
2 {
3     /**
4     * Standard constructor of the ToolProvider.
5     *
6     * @param DataConnector $dataConnector The data connector
7     * required by the LTI framework
8     */
9     public function __construct($dataConnector)
10    {
11        parent::__construct($dataConnector);
12    }
13
14    public function onLaunch()
15    {
16
17    }
18
19    public function onContentItem()
20    {
21
22    }
23
24    public function onRegister()
25    {
26
27    }
28
29    public function onError()
30    {
31
32    }
33 }
```

Listing 3.7: Extension of the ToolProvider base class

3.3.3 The LTI Outcome Service - The Grade Book

The quiz application enables teachers and instructors to send the grades that were created in the course of a quiz, back to the LTI consumer that initiated the quiz process. Every LTI consumer can have a grade book. To access this grade book, the calling LTI consumer must enable this feature, or otherwise, requests to the according service are blocked by the responsible LTI routines. The service that is accountable for handling incoming grade submissions is called LTI Outcome Service and the respective service URL is transmitted as part of every LTI launch request under the name "lis_outcome_service_url".

An example of the contents of an LTI launch request can be found in the appendix A.1. To make use of the grade book, two of the items that are incorporated in the launch request are particularly important: the formerly mentioned "lis_outcome_service_url" and secondly, the "lis_result_sourcedid". The first parameter of the launch request identifies the endpoint to which grades should be posted and the "sourcedID" parameter tells the outcome service, from which user and resource link, the grades originate from.

The PHP LTI framework already contains an Outcome class⁸ that is meant to target the described outcome service, but for some reason it was not possible to get the requests to work with this class. Therefore it was necessary to write a custom implementation, called "GradebookReplaceRequest" class, that enables the appropriate use of the LTI outcome service.

The custom outcome class uses CURL as tool for delivering the requests to the defined outcome service. The requests to the LTI services are XML requests that must have a specific format⁹. An example request is also included in the appendix under A.2. It is recommended to use the described grade book "replaceResult" request for every insertion or update of grades, which is the case for the custom implementation and the grade values that are transmitted must be within the floating-point range of zero to one, or will be rejected by the target outcome service.

⁸LTI Framework Documentation - Outcomes <https://github.com/MSGlobal/LTI-Tool-Provider-Library-PHP/wiki/Outcomes> 10.07.2019

⁹LTI Implementation Guide on Outcomes <https://www.imslobal.org/specs/ltiv1p1/implementation-guide#toc-6> 10.07.2019

3.3.4 Testing LTI Applications

When developing applications that support the LTI protocol, a representative testing environment might not be available at all times. To make the LTI launch requests for LTI tool providers functional, there is one tool that is very helpful in development: The LTI consumer emulator¹⁰ created by the ceLTic project¹¹. The emulator is capable of simulating LTI 1.0 and LTI 2.0 requests of all types, supporting all common attributes, for example, the LTI standard roles. It is also possible to set up custom parameters and edit every piece of information that is sent with the requests. Besides that, it also offers a grade book service, which can be used to test the LTI outcome service for grades and visualize the results. The emulator is also very useful for testing with multiple users, that might not be present in the original target LTI consumer and can very easily be created with the emulator by just switching the LTI user IDs.

Figure 3.24 shows how the emulator looks in its default state after it is opened. To use the emulator, only a few things must be set up before the first launch request can be directed towards an application under development. The top left box of the emulator provides three input fields, which are the most important for launching an LTI tool provider. Once the "Launch URL", "Consumer Key" and "Shared Secret" are set up, a meaningful request can be sent to the target LTI tool provider. The launch URL points to the entry script of the tool provider application and the shared secret and consumer key must match to what is set up in the tool provider configuration.

3.3.5 User Interface: Bootstrap, jQuery and jQueryUI

As stated before, the perceived usefulness of e-Learning systems is to some degree dependent on the perceived quality of the delivering system (Sun et al., 2008) and therefore it was a goal to make the user interface of the application appealing for the users. To achieve this, the famous Bootstrap framework¹² was used to create a graphically consistent and nice-looking

¹⁰LTI Consumer Emulator <http://ltiapps.net/test/tc.php> 05.07.2019

¹¹ceLTic Project <http://www.celtic-project.org/> 05.07.2019

¹²Bootstrap CSS Framework <https://getbootstrap.com> 06.07.2019

3 Development of a Quiz

The screenshot shows a web browser window with the URL `http://lapps.net/test/lc.php`. The page header includes the **ceLTIc** logo and the text "Creating Environments for Learning using Tightly Integrated Components formerly JISC-funded project". Below the header is a section titled "IMS LTI Tool Consumer emulator" with a brief description. A blue banner for "SALTIRE" asks for feedback. The main interface consists of several form panels:

- Registration settings:** Includes fields for Launch URL, Consumer key, and Shared secret, each with a "[R]" icon.
- Message data:** Includes Message type, LTI version, Target frame, Width, Height, Return URL, and Signature method.
- Tool consumer data:** Includes GUID, Name, Contact email, Product family, and Version.
- Context/resource data:** Includes Context ID, Context title, Context label, Resource ID, and Resource title.
- User data:** Includes User ID, Given name, Family name, Full name, Email address, Roles, and SUIser.username.

Figure 3.24: The LTI consumer emulator by the ceLTIc project

user experience. Every view in the quiz application uses the bootstrap grid system, which erased the need to take care of floating and clearing floats in CSS. The grid system also plays a big role in creating consistent layouts throughout an application.

To increase the responsiveness of the system and provide instantaneous feedback on user actions, the free JavaScript DOM-Manipulation library jQuery was used and to increase the reusability of more complex visual elements, such as modal windows, draggable elements, or sortable lists it was supplemented by the jQueryUI¹³ extension, which provides a comprehensive collection of complex user interface components.

3.3.6 Database

The database that was used for the development of the quiz application is MariaDB in the version 10.1.37. MariaDB uses the same syntax as the very successful MySQL-Database and is, except for a few cases, fully compatible

¹³jQueryUI Library <https://jqueryui.com> 06.07.2019

3 Development of a Quiz

¹⁴ to MySQL and can be used as a replacement for it. See the database model in figure 3.25. In the diagram, the tables used for the roles and permissions, as well as the tables related to the LTI framework were omitted to maintain clarity and focus on the custom implementation of the quiz application.

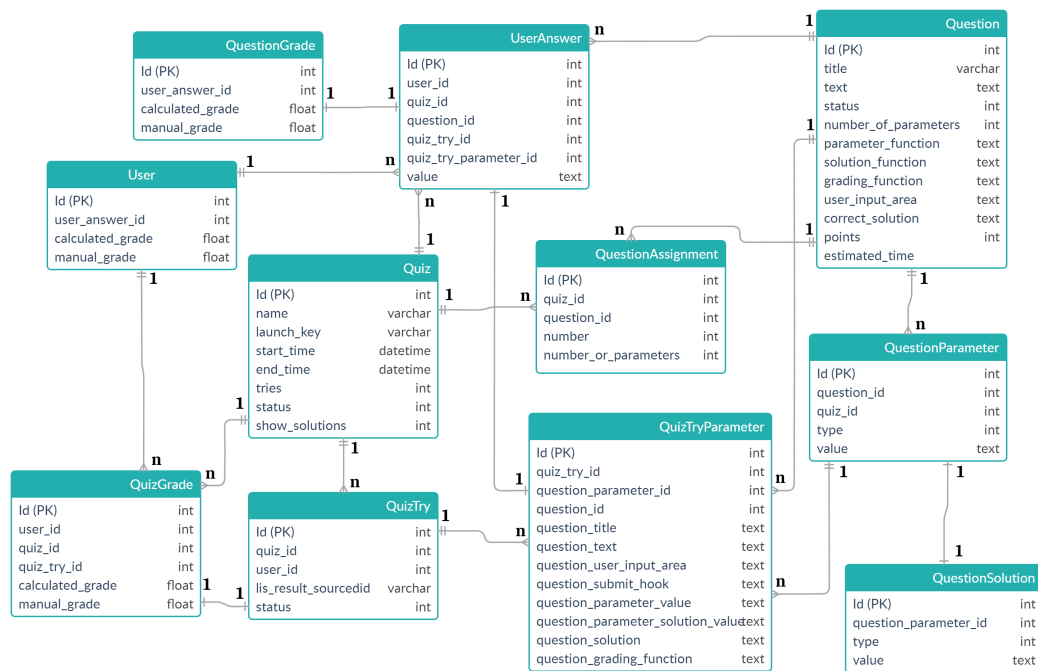


Figure 3.25: Database model of the quiz application

Data Model

The data model revolves around questions and quizzes. Questions are assigned to quizzes with the help of the question assignment junction table, which creates an entry for every question that is assigned to a quiz and defines the order within a quiz for the questions. When creating questions, so many entries in the question parameter table are inserted, as the number of preview parameters defined on the question page. In the same turn,

¹⁴MariaDB to MySQL Compatibility <https://mariadb.com/kb/en/library/mariadb-vs-mysql-compatibility> 07.07.2019

3 Development of a Quiz

whenever a question parameter is generated, a question solution is generated as well, containing the calculated solution values in JSON form for every question parameter. The question parameters can have two different types that identify them as either a question or as a quiz parameter.

In the quiz creation process, question parameters of the quiz parameter type are created. The reason for this behaviour is to separate the preview parameters for questions, from the ones that are assigned to a quiz, which enables the creator to define the amount of parameters in the quiz creation process rather than at the time of creating questions, because at the quiz creation stage the teacher is more likely to know how many students will participate in a quiz.

When a new quiz try is created, one parameter is selected randomly from the quiz parameter set of the quiz for each question. This selection is associated with the newly created quiz try and all information about the content generators is stored in the just created, so-called "Quiz Try Parameter". The reason for this is that the editing of questions could invalidate existing user results and cause malfunction of the grading interface, when for example the values produced in Q , do not match the values produced by a later modified $S(Q)$. In such a case automated grading would become impossible, thus a snapshot of all relevant information is stored on each quiz try, to ensure that grading will be possible in the future.

When it comes to grades, the database is set up to differentiate between quiz and question grades with the respective tables. The main subject of the grading process is the user answer table, which is connected to a quiz try parameter. As the application remembers the parameters and question information at the time of the quiz attempt, these values are used to calculate the grades. This means, that on the grading page, the grading function is taken from the quiz try parameter table and not from the actual question, to not confuse potentially different versions of these functions. The results are stored per user and per question as question grades. The final product of the quiz process is a quiz grade, which is calculated as the average of the relevant question grades.

The user table that represents either an application user or an LTI user, is connected to one or multiple roles by a junction table (which is not shown

3 Development of a Quiz

in the data model). Permissions are also connected to roles with the help of a junction table.

Doctrine ORM

Zend Framework comes with its own built-in database abstraction mechanism, called "Zend-DB", that requires the user to write a lot of code, especially when it comes to complex join statements. To make it easier to work with relational data, Doctrine ORM was used.

The Doctrine Object Relationship Mapper¹⁵ abstracts the database layer and enables a programmer to use convenient functions to handle the data more easily within the application. Doctrine is built to incorporate object relations in so-called entity classes, that are created in the application. Every entity must define all of its relations to other entities as described in the Doctrine Association Mapping Documentation¹⁶. After they are configured properly, it is very efficient to work with the doctrine relations.

The concept behind Doctrine is based on two design patterns called "Data Mapper" and "Unit of Work", where the Data Mapper is responsible for retrieving and saving records from and to the database, while the Unit of Work pattern remembers the states of all objects managed by the Data Mapper. This means that the application itself has no need to know anything about the database system that is used and relies solely on the use of the created entity classes. (Dunglas, 2013)

3.3.7 Security Considerations

It was mentioned several times before that the quiz application protects all actions with a role-based access control mechanism and that it was taken from the free open source book Oleg (2018). Together with the LTI

¹⁵Doctrine ORM <https://www.doctrine-project.org> 07.07.2019

¹⁶Doctrine Association Mapping Documentation <https://www.doctrine-project.org/projects/doctrine-orm/en/2.6/reference/association-mapping.html> 07.07.2019

3 Development of a Quiz

authentication of the users, this is considered the main security mechanism of the quiz application.

The LTI authentication uses the OAuth 1.0 Body Signing¹⁷ principle. In this procedure an authentication header is applied to the posted HTTP-Request. This header includes a hash of the whole message that is calculated with an OAuth consumer key and a shared secret. Both ends, the sender as well as the receiver, have to know the shared secret to verify the authenticity and integrity of the message. In the quiz application, this is done by the used LTI framework and can be considered as the security base layer of the application. If the shared secret and consumer key are known, an attacker that knows how to construct LTI requests, can launch the application and impersonate users as he desires. Therefore the consumer key and shared secret must be handled with care and should be of appropriate complexity.

As long as the LTI communication can be considered secure, the rest of the application is protected by the RBAC system, which only allows logged in users to take any actions and further guarantees that on every action, a user can only perform it, when he or she has access to the functionality as defined by the application permissions.

Measures against Cross-Site-Scripting Attacks

When users may input data that is later rendered in the application, the possibility to commit Cross-Site-Scripting (XSS) attacks arises. The OWASP standard cheat sheet¹⁸ states that it is not enough to just HTML Entity Encode the input, but to successfully prevent XSS attacks, the syntax must be encoded depending on where in a HTML document the untrusted data is rendered.

When rendering the user input in views, each part that comes from the user answer fields in the front-end of the application is escaped with the use of

¹⁷OAuth 1.0 Body Signing <https://oauth1.wp-api.org/docs/basics/Signing.html>
07.07.2019

¹⁸OWASP XSS Attack Cheat sheet https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.md
28.06.2019

3 Development of a Quiz

the Zend Framework Escaper class¹⁹.

The quiz-attempt-page in the front-end, posts the given user answers to the according controller in the form of a serialized HTML form. This form could be altered by hand and contain not only malicious values for user answers but also malicious name attributes. There are three places where the user answers are actually rendered on a page: the solutions page, the question grade generation page and the question grade review page. In all occurrences of the user answers in views, they are either encoded as JSON, which disables all HTML and particularly script tags by escaping the slashes, or in the case of the solutions page. They are escaped with the Zend Escaper HTML-Escaping functions. In the case of the potentially dangerous input being only present as JavaScript JSON object, it is also handled as a string, so unless the input is executed in a manner that is equal to the JavaScript eval() function, no injected code can be executed.

Another common attack vector for XSS attacks, are GET-Parameters that are rendered in views. This is not the case at any point of the quiz application.

Measures against SQL Injections

To prevent the very common attack vector of SQL Injections, every database operation in the application (except in one place) uses the Doctrine Entity Manager's²⁰ **find()** and **persist()** methods, to retrieve and persist records to the database. As stated in the Doctrine documentation for security²¹ this is a sufficient countermeasure against SQL Injections. The only code where a different method is used is the filter function for active questions on the quiz creation page. In this case, the Doctrine QueryBuilder was used to implement a meaningful search, which is by default not protected against injections but offers methods to use it with prepared statements and

¹⁹Escaping HTML with the Zend Escaper <https://docs.zendframework.com/zend-escaper/escaping-html> 29.06.2019

²⁰Doctrine Entity Manager API Reference <https://www.doctrine-project.org/api/orm/latest/Doctrine/ORM/EntityManager.html> 28.06.2019

²¹Doctrine ORM Documentation - Security <https://www.doctrine-project.org/projects/doctrine-orm/en/2.6/reference/security.html> 28.06.2019

3 Development of a Quiz

set the parameters for these statements. In this case, the Doctrine security documentation considers the use of the QueryBuilder²² safe.

²²Doctrine QUeryBuilder API Reference <https://www.doctrine-project.org/api/orm/latest/Doctrine/ORM/QueryBuilder.html> 28.06.2019

4 Discussion

This chapter elaborates on some controversial points in connection with this thesis.

The quiz application that was developed in the course of this master thesis, has a very specific requirement. The creators of the quizzes should be able to program questions with the full support of JavaScript and HTML. This creates a myriad of problems by giving the creators of the questions a lot of freedom which also creates room for errors. There is also a fine line between cutting down the possibilities of the technologies while providing an easy to use user interface, without limiting the options for creative question implementations too much.

The application tries to provide as much feedback in the question creation process as possible, but fails at some occasions. One problem is the evaluation of the correctness of teacher-provided JavaScript code. Whenever, for example, question parameters are generated, this happens on a separate page which executes the JavaScript code of the Question Parameter Generator Function: Q. This happens within a JavaScript Try-Catch block and catches all catchable errors. Unfortunately, syntax errors do not count as such errors and break the page. To avoid this kind of errors and provide a better user experience and clearness, it would be good to validate the syntax of the provided expressions beforehand, but since this is complicated to implement it was omitted. This puts the creator of the question in a state where he or she needs to carefully debug the code in the parameter generation page, which is arguably more complicated than debugging the same code on a plain page, where only the instructor's code is present. The question parameter generation page includes a lot of JavaScript code that is needed to generate the defined amount of parameters.

4 Discussion

The user input area also has limitations due to the differences in the structure of HTML input fields of different types. A checkbox field value cannot be set in the same way as the value of a text field. This is also true for other input types, such as radio buttons or input selections. It would be necessary to parse the input type of every input to prepopulate the field values for given answers when a quiz attempt is resumed. The application therefore only supports the automatic population of given answers in unfinished and reloaded quiz attempts for the HTML input types text and hidden. When other input types are used in the question answer form, it would be possible to use hidden input fields to load the values. Thereafter, it is necessary to include custom JavaScript code that analyzes the value in the hidden field and sets the state of the corresponding input field as desired.

Moodle provides more than only one mode for the grading of its quizzes. The implementation of the grading interface computes the grades for all quiz attempts of all students, but only shows the last closed attempt to the teachers. This makes possible future implementations for different modes such as "highest grade" easier as the grade data is already present in the system, but it is data that is not used at this moment.

Another point that should be mentioned, is that a custom implementation of a quiz is generally a duplication of Moodle's built-in quiz engine. Even though, thanks to the LTI protocol, the tool is generally reusable, it defeats the consistency of Moodle quizzes and most likely has a custom design and behavior. Furthermore, it is a lot of work to provide even a small subset of the quiz features that Moodle implements and the development and maintenance efforts should be kept in mind. The first approach should always be to evaluate carefully the possibilities that are already present in Moodle and its plugins.

The last issue that must be mentioned is the used version of the LTI protocol. The target Moodle environment of the quiz application was using the LTI protocol of the version 1.0, while a newer version (2.0) already existed. The LTI framework that was used to create the application generally supports the new version and functions out of the box, but that does not mean that an LTI 1.0 compatible application functions automatically as intended. This depends on the requests used by the tool consumer and might require minor adaptations on the LTI connection process and outcome management.

5 Outlook

This chapter describes some ideas that evolved during the implementation of the quiz application over the course of creating this thesis.

5.1 Gamification

The implementation of gamification elements in learning environments is an interesting topic that could improve learning results. Gamification describes a process that takes an existing serious system and equips it with elements that are usually found in games.

Buckley and Doyle (2016) investigated the effects of gamification in e-Learning environments. Even though the results might not be accurate and further research is needed to prove the positive impact of gamification intervention in learning environments, the study shows an improvement of outcomes, if gamification is used.

It would be interesting to investigate whether the use of gamification elements, such as badges, levels, leader boards or experience points, would result in higher overall grades and increased student engagement. There are plugins available for Moodle ¹, that incorporate gamification elements, such as badges, leveling mechanics, or funny animations into the learning environment. Therefore, it should be carefully evaluated if an implementation of gamification elements would be viable in the quiz application itself, rather than using the available Moodle plugins.

¹Moodle Gamification Plugins <https://moodle.org/plugins/browse.php?list=set&id=88> 29.07.2019

5.2 Learning Analytics

As described in this master thesis, a vital part of the teaching and learning process, is the generation of knowledge about the information delivery quality and quality of the used materials, with the intention of improving the teaching method. Moodle provides some useful learning analytics features² out of the box. For example, it is possible to set up notifications about learners who interact poorly with the system or are in danger of dropping out. When configured properly such a system can be used with the grades produced by the homework quizzes, provided by the quiz application.

One issue that cannot be analyzed with the Moodle features by any means, are the questions themselves. Data about how long students need to complete a given question or quiz, can only be gathered by the quiz application and thus it would be an interesting feature to implement in the future. Such data could be used to minimize the divergence between the perceived difficulty of questions between a teacher and the students and provide a more rewarding learning environment. It could reveal problems, such as unclarities in question texts or other misunderstandings that might be concealed, but influence the grades of the students and might decrease engagement.

5.3 Evaluation Study

Some interesting studies were conducted by various authors about the effects of the application of e-Learning technology to classic courses (see 2.2.1).

The study by Stan and Manea (2015) was particularly interesting and comparable to the hybrid environment present for the quiz application. It would be interesting to monitor if there are any improvements in the final exam pass rates compared to the time before the quiz application was used for smaller self-assessment exercises during the semester.

²Moodle 3.7 Learning Analytics Documentation <https://docs.moodle.org/37/en/Analytics> 25.07.2019

6 Conclusion

The outcome of this master thesis, an implementation of a quiz application in which teachers can create questions that contain HTML and JavaScript and which can be integrated into Moodle, was successful. Open standards, such as the LTI protocol are very important and a great help in providing reusable and, more importantly, shareable learning resources. While implementing the features of the quiz, it became clear that when retries are involved, the flow of procedures and resulting states of a quiz, were more complex than initially thought. Especially retries combined with randomized question parameters for the questions, demanded a more complex database design as defined at the beginning of the implementation, as questions about the persistent storage of results arose. This is problematic when a question is modified at the time when a quiz is taken. This could potentially invalidate the existing results. Therefore, it was necessary to duplicate the information and store all question information along with each quiz try of every user, which was not intended at the beginning.

Another insight is, that to provide a meaningful online learning experience for both, teachers and students, the quality of the application should be high. Modern leading systems in the field, set very high standards and the perceived quality of an application matters in the e-Learning process, which makes any implementation more demanding.

The positive effects of the application of e-Learning technology and especially self-assessment strategies, seem to be widely accepted by the experts of the field and the spread of the use of e-Learning technology is growing worldwide. Hopefully, this movement towards more open and borderless access to educational resources (as for example with MOOCs) can lead to a better world, or at least change the world of education for the better. Especially self-assessment strategies combined with openly accessible learning

6 Conclusion

resources could provide an economically sustainable worldwide education system.

6 Conclusion

Glossary

CSS Cascading Style Sheets. Technology and language designed to alter the visualization of HTML and DOM documents.

CURL Client for URLs. Program library used to transmit data over the internet.

DOM Document Object Model. Specification for a tree-based structure of HTML documents.

HTML Hypertext Markup Language. Scripting language used to build web applications.

JavaScript Client-side scripting language for web applications.

JSON JavaScript Object Notation. Method to encode information, suitable for web-applications.

LMS Learning Management System. System used to create and serve online learning materials.

LTI Learning Tools Interoperability. Protocol used to connect e-Learning Systems.

LTI Grade Book An LTI Outcome Service that can receive and store grades from LTI tool providers.

LTI Tool Consumer LTI compatible applications that consume other LTI applications and present them as part of themselves.

LTI Tool Provider LTI compatible application that is started from a different system.

MOOC Massive Open Online Course. Educational platforms with openly accessible courses for large numbers of students.

Moodle Popular open source Learning Management System.

Glossary

- MVC** Model View Controller. Software design pattern commonly used in web application development. Separates database modifications, program logic and visual representation.
- ORM** Object Relationship Mapper. Database abstraction technology for easier use of data in application programming.
- PHP** Hypertext Preprocessor. Server-side scripting language for web applications.
- SQL Injection** Structured Query Language Injection. Common attack against web applications with the goal of data manipulation or service disruption.
- UI** User Interface. Graphical representation of computer programs.
- URL** Uniform Resource Locator. Identification for network resources and commonly used to target resources in the internet.
- XML** Extensible Markup Language. Flexible document format used to transfer structured information between machines.
- XSS Attack** Cross-Site-Scripting Attack. Attack that injects code into a web page displayed to a user.

Appendices

A.1 Example LTI Launch Request

```
1 Array
2 (
3     [oauth_version] => 1.0
4     [oauth_nonce] => d7d6193c85bba38476e28052e3e1719e
5     [oauth_timestamp] => 1562772804
6     [oauth_consumer_key] => consumer12345
7     [lti_message_type] => basic-lti-launch-request
8     [lti_version] => LTI-1p0
9     [resource_link_id] => 429785226
10    [resource_link_title] => Phone home
11    [resource_link_description] => Will ET phone home, or not;
    click to discover more.
12    [user_id] => 29123
13    [roles] => Instructor
14    [lis_person_name_full] => John Logie Baird
15    [lis_person_name_family] => Baird
16    [lis_person_name_given] => John
17    [lis_person_contact_email_primary] => jbaird@uni.ac.uk
18    [lis_person_sourcedid] => sis:942a8dd9
19    [user_image] => http://ltiapps.net/test/images/lti.gif
20    [context_id] => S3294476
21    [context_type] => CourseSection
22    [context_title] => Telecommunications 101
23    [context_label] => ST101
24    [lis_course_offering_sourcedid] => DD-ST101
25    [lis_course_section_sourcedid] => DD-ST101:C1
26    [tool_consumer_info_product_family_code] => jisc
27    [tool_consumer_info_version] => 1.2
28    [tool_consumer_instance_guid] => vle.uni.ac.uk
29    [tool_consumer_instance_name] => University of JISC
30    [tool_consumer_instance_description] => A Higher Education
    establishment in a land far, far away.
31    [tool_consumer_instance_contact_email] => vle@uni.ac.uk
32    [tool_consumer_instance_url] => https://vle.uni.ac.uk/
33    [launch_presentation_return_url] => http://ltiapps.net/test/tc-
    -return.php
34    [launch_presentation_css_url] => http://ltiapps.net/test/css/
    tc.css
35    [launch_presentation_locale] => en-GB
36    [launch_presentation_document_target] => frame
37    [lis_outcome_service_url] => http://ltiapps.net/test/tc-
    outcomes.php
```

```

38 [lis_result_sourcedid] => 5do209eb1383eob188fbf560126b567c:::
S3294476:::29123:::dyJ86SiwwA9
39 [ext_ims_lis_basic_outcome_url] => http://ltiapps.net/test/tc-
ext-outcomes.php
40 [ext_ims_lis_resultvalue_sourcedids] => decimal
41 [ext_ims_lis_memberships_url] => http://ltiapps.net/test/tc-
ext-memberships.php
42 [ext_ims_lis_memberships_id] => 5
do209eb1383eob188fbf560126b567c:::4jflkkdf9s
43 [ext_ims_lti_tool_setting_url] => http://ltiapps.net/test/tc-
ext-setting.php
44 [ext_ims_lti_tool_setting_id] => 5
do209eb1383eob188fbf560126b567c:::d94gjklf954kj
45 [custom_tc_profile_url] => http://ltiapps.net/test/tc-profile.
php/5do209eb1383eob188fbf560126b567c
46 [custom_system_setting_url] => http://ltiapps.net/test/tc-
settings.php/system/5do209eb1383eob188fbf560126b567c
47 [custom_context_setting_url] => http://ltiapps.net/test/tc-
settings.php/context/5do209eb1383eob188fbf560126b567c
48 [custom_link_setting_url] => http://ltiapps.net/test/tc-
settings.php/link/5do209eb1383eob188fbf560126b567c
49 [custom_lineitems_url] => http://ltiapps.net/test/tc-outcomes2
.php/5do209eb1383eob188fbf560126b567c/S3294476/lineitems
50 [custom_results_url] => http://ltiapps.net/test/tc-outcomes2.
php/5do209eb1383eob188fbf560126b567c/S3294476/lineitems/
dyJ86SiwwA9/results
51 [custom_lineitem_url] => http://ltiapps.net/test/tc-outcomes2.
php/5do209eb1383eob188fbf560126b567c/S3294476/lineitems/
dyJ86SiwwA9
52 [custom_result_url] => http://ltiapps.net/test/tc-outcomes2.
php/5do209eb1383eob188fbf560126b567c/S3294476/lineitems/
dyJ86SiwwA9/results/29123
53 [custom_context_memberships_url] => http://ltiapps.net/test/tc-
memberships.php/context/5do209eb1383eob188fbf560126b567c
54 [custom_link_memberships_url] => http://ltiapps.net/test/tc-
memberships.php/link/5do209eb1383eob188fbf560126b567c
55 [oauth_callback] => about:blank
56 [oauth_signature_method] => HMAC-SHA1
57 [oauth_signature] => FyimedCPjVURqMpA9EKaQpc+T64=
58 )

```

Listing 1: Example POST of an LTI Launch Request

A.2 Outcome Service - Replace Result Request XML

```
1 <?xml version = "1.0" encoding = "UTF-8"?>
2
3 <imsx_POXEnvelopeRequest xmlns = "http://www.imsglobal.org/
4   services/ltiv1p1/xsd/imsoms_v1p0">
5   <imsx_POXHeader>
6     <imsx_POXRequestHeaderInfo>
7       <imsx_version>V1.0</imsx_version>
8       <imsx_messageIdentifier>999999123</imsx_messageIdentifier>
9     </imsx_POXRequestHeaderInfo>
10    </imsx_POXHeader>
11    <imsx_POXBody>
12      <replaceResultRequest>
13        <resultRecord>
14          <sourcedGUID>
15            <sourcedId>3124567</sourcedId>
16          </sourcedGUID>
17          <result>
18            <resultScore>
19              <language>en</language>
20              <textString>0.92</textString>
21            </resultScore>
22          </result>
23        </resultRecord>
24      </replaceResultRequest>
25    </imsx_POXBody>
26  </imsx_POXEnvelopeRequest>
```

```
40
41     </resultRecord>
42
43     </replaceResultRequest>
44
45     </imsx_POXBody>
46
47 </imsx_POXEnvelopeRequest>
```

Listing 2: Example XML Grade ReplaceRequest

Bibliography

- Agarwal, Anant (2018). *Updates To Our Platform: Achieving Long-Term Sustainability*. URL: <https://blog.edx.org/updates-platform-achieving-long-term-sustainability> (visited on 07/09/2019) (cit. on p. 9).
- Alsabawy, Ahmed Younis, Aileen Cater-Steel, and Jeffrey Soar (2016). "Determinants of perceived usefulness of e-learning systems." In: *Computers in Human Behavior* 64, pp. 843–858. ISSN: 07475632. DOI: [10.1016/j.chb.2016.07.065](https://doi.org/10.1016/j.chb.2016.07.065) (cit. on p. 42).
- Amo, Elisa and Francisco Jareño (2011). "Self, Peer and Teacher Assessment as Active Learning Methods." In: *Research Journal of International Studies* 18.18, pp. 41–47. URL: http://www.uclm.com/profesorado/fjareno/DOCS/RJIS%7B%5C_%7D18%7B%5C_%7D05.pdf (cit. on p. 6).
- Barbosa, Héctor and Francisco Garcia (2005). "Importance of online assessment in the E-learning process." In: *ITHET 2005: 6th International Conference on Information Technology Based Higher Education and Training, 2005*. Vol. 2005. ISBN: 0780391411. DOI: [10.1109/ITHET.2005.1560287](https://doi.org/10.1109/ITHET.2005.1560287) (cit. on pp. 4, 5).
- Buckley, Patrick and Elaine Doyle (2016). "Gamification and student motivation." In: *Interactive Learning Environments* 24.6, pp. 1162–1175. DOI: [10.1080/10494820.2014.964263](https://doi.org/10.1080/10494820.2014.964263). eprint: <https://doi.org/10.1080/10494820.2014.964263>. URL: <https://doi.org/10.1080/10494820.2014.964263> (cit. on p. 67).
- Ćukušić, Maja, Željko Garača, and Mario Jadrić (2014). "Online self-assessment and students' success in higher education institutions." In: *Computers and Education* 72, pp. 100–109. ISSN: 03601315. DOI: [10.1016/j.compedu.2013.10.018](https://doi.org/10.1016/j.compedu.2013.10.018) (cit. on pp. 1, 7, 42).
- Dunglas, Kévin (2013). *Persistence in PHP with Doctrine ORM*. Packt Publishing Ltd. ISBN: 978-1-78216-410-4 (cit. on p. 61).

Bibliography

- Ebner, Martin (2019). "Virtuelle Lernorte: eine Übersicht." In: URL: https://www.researchgate.net/publication/331981005_Virtuelle_Lernorte_eine_Ubersicht (cit. on pp. 8, 11, 12).
- Harandi, Safiyeh Rajaei (2015). "Effects of e-learning on Students' Motivation." In: *Procedia - Social and Behavioral Sciences* 181, pp. 423–430. ISSN: 18770428. DOI: [10.1016/j.sbspro.2015.04.905](https://doi.org/10.1016/j.sbspro.2015.04.905) (cit. on pp. 1, 7).
- Harasim, Linda (2000). "Shift happens: Online education as a new paradigm in learning." In: *Internet and Higher Education* 3.1-2, pp. 41–61. ISSN: 10967516. DOI: [https://doi.org/10.1016/S1096-7516\(00\)00032-4](https://doi.org/10.1016/S1096-7516(00)00032-4). URL: <http://www.sciencedirect.com/science/article/pii/S1096751600000324> (cit. on p. 3).
- Khalil, Mohammad and Martin Ebner (2016). "When Learning Analytics Meets MOOCs - a Review on iMooX Case Studies." In: vol. 648, pp. 3–19. ISBN: 978-3-319-49465-4. DOI: [10.1007/978-3-319-49466-1_1](https://doi.org/10.1007/978-3-319-49466-1_1) (cit. on p. 11).
- Liu, Ngar Fun and David Carless (2006). "Peer feedback: The learning element of peer assessment." In: *Teaching in Higher Education* 11.3, pp. 279–290. ISSN: 13562517. DOI: [10.1080/13562510600680582](https://doi.org/10.1080/13562510600680582) (cit. on p. 4).
- Malala Yousafzai, Christina Lamb (2013). *I Am Malala: The Story of the Girl Who Stood Up for Education and Was Shot by the Taliban*. Weidenfeld & Nicolson. ISBN: 3642253636 (cit. on p. 1).
- Nicol, David and Debra MacFarlane-Dick (2006). "Formative assessment and selfregulated learning: A model and seven principles of good feedback practice." In: *Studies in Higher Education* 31.2, pp. 199–218. ISSN: 03075079. DOI: [10.1080/03075070600572090](https://doi.org/10.1080/03075070600572090) (cit. on p. 6).
- Nielsen, Jakob (2007). *Breadcrumb Navigation Increasingly Useful*. URL: <https://www.nngroup.com/articles/breadcrumb-navigation-useful> (visited on 06/30/2019) (cit. on p. 20).
- Oleg, Krivtsov (2018). *Using Zend Framework 3*. URL: <https://olegkrivtsov.github.io/using-zend-framework-3-book/html/index.html> (cit. on pp. 17, 50, 53, 61).
- Palmer, Michele (2015). *87% of Students Say They Gain as Much or More From Online Courses Compared to On-campus Courses*. URL: <https://blog.edx.org/87-students-say-they-gain-much-or-more> (visited on 07/09/2019) (cit. on p. 12).
- Richey, Rita C., Kenneth H. Silber, and Donald P. Ely (2008). "Reflections on the 2008 AECT Definitions of the Field." In: *TechTrends* 52.1, pp. 24–

Bibliography

25. ISSN: 1559-7075. DOI: [10.1007/s11528-008-0108-2](https://doi.org/10.1007/s11528-008-0108-2). URL: <https://doi.org/10.1007/s11528-008-0108-2> (cit. on p. 3).
- Stan, Cristian and Adriana Denisa Manea (2015). "The Divergent Relationship between Assessment and Self-assessment in Higher Education. Experimental Results." In: *Procedia - Social and Behavioral Sciences* 209, pp. 497–502. ISSN: 18770428. DOI: [10.1016/j.sbspro.2015.11.278](https://doi.org/10.1016/j.sbspro.2015.11.278) (cit. on pp. 1, 7, 68).
- Sun, Pei Chen et al. (2008). "What drives a successful e-Learning? An empirical investigation of the critical factors influencing learner satisfaction." In: *Computers and Education* 50.4, pp. 1183–1202. ISSN: 03601315. DOI: [10.1016/j.compedu.2006.11.007](https://doi.org/10.1016/j.compedu.2006.11.007) (cit. on pp. 42, 57).
- Valenti, Salvatore, Alessandro Cucchiarelli, and Maurizio Panti (2002). "Computer Based Assessment Systems Evaluation via the ISO9126 Quality Model." In: *Journal of Information Technology Education: Research* 1, pp. 157–175. ISSN: 1547-9714. DOI: [10.28945/353](https://doi.org/10.28945/353). URL: <http://www.jite.informingscience.org/documents/Vol1/v1n3p157-175.pdf> (cit. on p. 5).
- Vardi, Moshe Y. (2012). "Will MOOCs destroy academia?" In: *Communications of the ACM* 55.11, pp. 5–5. ISSN: 00010782. DOI: [10.1145/2366316.2366317](https://doi.org/10.1145/2366316.2366317). URL: <http://dl.acm.org/citation.cfm?doid=2366316.2366317> (cit. on p. 12).
- Vickers, Stephen P. and Simon Booth (2014). *Learning Tools Interoperability® (LTI®): A Best Practice Guide*. URL: http://ltiapps.net/guide/LTI_Best_Practice.pdf (cit. on p. 53).
- Zimmerman, Barry J. (1990). "Self-Regulated Learning and Academic Achievement: An Overview." In: *Educational Psychologist* 25.1, pp. 3–17. ISSN: 15326985. DOI: [10.1207/s15326985ep2501_2](https://doi.org/10.1207/s15326985ep2501_2) (cit. on p. 5).