



Norwegian University of
Science and Technology

Peer Review Module Development in the Blackboard Learning Management System

Anders Lunde

Robin Singh Sjøvoll

Master of Science in Computer Science

Submission date: June 2017

Supervisor: Trond Aalberg, IDI

Norwegian University of Science and Technology
Department of Computer Science

Abstract

In present education, a range of assessment methodologies are being used. From standard exercise deliveries to larger projects incorporating many sub-processes. With the introduction of Learning Management Systems (LMS), these methodologies became much more accessible. And by having a LMS that can be utilized as a development platform one is presented with a lot of possibilities, like being able to tailor the educational process through the use of digitized LMS content. This can in turn potentially enhance the learning outcome for the students.

One such assessment methodology is peer reviewing. And through analyzing state of the art software, researching on theoretical aspects closely related to peer reviewing, and conducting semi-structured interviews as part of a qualitative data gathering, we were able to identify important features and functionality which we used to design and develop a digital peer reviewing module.

This module was then integrated into Blackboard, a LMS that was about to be introduced at NTNU, by following an experimental development paradigm. The main focus was to map the overall potential of Blackboard as a development platform by identifying bottlenecks and opportunities with Blackboards development strategies, focusing on an implementation strategy called building blocks.

The main results of this master thesis can be split in two parts; The developed peer review prototype, including extended features like evaluations of reviews and process streamlining. And the developer experiences, a detailed summary for future developers to capitalize on in order to make more educational development choices.

Sammendrag

I dagens utdanning finnes det en rekke forskjellige vurderingsmetoder. Fra standard oppgaveleveringer til større prosjekter som inneholder og tar i bruk flere sub-prosesser. Og med introduksjonen til digitale læringsplattformer, ble disse vurderingsmetodene mye lettere tilgjengelig. Ved å ha digitale læringsplattformer som også kan bli benyttet som utviklingsplattform åpner man opp for mange muligheter, da dette gjør det mulig å skreddersy utdanningsprosessen gjennom digitalisert innhold og dermed potensielt forbedre læringssutbyttet til studentene.

Et eksempel på en slik vurderingsmåte er medstudentevaluering. Og gjennom å analysere state of the art programvare, undersøke teoretiske aspekter som er nært knyttet til medstudentevaluering som et konsept, og gjennomføre semi-strukturerte intervjuer som en del av en kvalitativ datasamling, klarte vi å identifisere viktige egenskaper og funksjonaliteter som ble brukt til å designe og utvikle en digital medstudentevaluering modul.

Denne modulen ble så integrert inn i Blackboard, et digitalt læringssystem som skulle bli tatt i bruk på NTNU, ved å følge et eksperimentelt forskningsparadigme. Hovedfokuset lå på å kartlegge det overordnede potensialet med Blackboard som en utviklingsplattform gjennom å identifisere flaskehalser og muligheter med forskjellige utviklingsstrategier mot Blackboard, og da med et spesielt fokus på en implementasjonsstrategi kalt building blocks.

Hovedresultatet i denne masteroppgaven kan deles opp i to deler; En digital prototype modul av medstudentevaluering, som inkluderer utvidede egenskaper som for eksempel evaluering av evaluering og effektivisering av oppgaveprosessen. Og erfaringene som ble tilegnet gjennom utviklingen mot Blackboard, presentert på en organisert og detaljert måte slik at fremtidige utviklere skal kunne benytte dem for å effektivisere sin egen utviklingsprosess.

Preface

This thesis is the result of research, design and development conducted in the Department of Computer and Information Science (IDI) at the Norwegian University of Science and Technology (NTNU). The work is part of the Computer Science study program and was conducted in the spring of 2017.

Our field of study is Software Engineering, thus we thought it would be interesting to develop some kind of software that could create some value for others. And with the introduction of a Learning Management System called Blackboard at NTNU, the opportunity to utilize it as a development platform was something that caught our eye. As having a LMS that can be tailored to the needs of an institution and the end-users is something that can be very powerful in an educational context.

The big question was then, what should be developed as a part of testing out the development platform? The choice fell on peer review, an assessment method mostly used in academic research to evaluate research papers. The method is also used in education, but to a much smaller degree. Thus, we thought that by designing and developing a simplistic and effective module that was easily accessible through the university LMS, more courses would try it out.

We would like to thank Trond Aalberg, our supervisor for all the guidance through discussions and feedback on both the development and the thesis itself, in addition to introducing us to such a relevant and important project subject. We would also like to thank Per Ivar Skinderhaug for all help with the development through giving us access to the Blackboard virtual machine, guiding us to documentation, and connecting us to relevant people.

Table of Contents

Abstract	i
Sammendrag	ii
Preface	iii
Table of Contents	vii
List of Figures	x
Abbreviations	xi
1 Introduction	1
1.1 Background and Motivation	1
1.2 Project Details	3
1.2.1 Description	3
1.2.2 Approach	3
1.3 Thesis Outline	4
2 Background	7
2.1 Peer review	7
2.1.1 Peer review in education	8
2.1.2 Peer review practises	9
2.1.3 Effective implementation and possible limitations	9
2.2 Peer review in a pedagogical setting	10
2.2.1 Theoretical aspects	10
2.2.2 Peer review as part of a pedagogical strategy	12
2.2.3 Educational support in LMS	13
2.3 Motivation	14
2.3.1 Motivation - an essential aspect	14
2.3.2 Peer review - Intrinsic or extrinsic motivation?	15

2.3.3	Goals as guiding criteria	15
2.3.4	Social effects	15
3	State of the art	17
3.1	Peer review in Learning Management Systems	17
3.1.1	Blackboard	17
3.1.2	Canvas	19
3.1.3	Google Classroom	20
3.1.4	ItsLearning	21
3.2	Third party peer review software	22
3.2.1	PeerMark	22
3.2.2	PRAZE	23
3.2.3	CPR	23
4	Interviews	25
4.1	The interview process	25
4.1.1	Type of interview	25
4.1.2	Interview questions	26
4.2	Interview results	27
4.2.1	Summary - Identified requirements	31
5	Design	33
5.1	Design Goals	33
5.2	Stakeholders	35
5.2.1	Stakeholder concerns	35
5.2.2	Target group	35
5.3	User interface	36
5.3.1	Guidelines	36
5.3.2	Blackboard theme	37
5.3.3	Chosen interface	38
5.4	Requirement specification	43
5.4.1	Functional requirements	43
5.4.2	Quality Requirements	48
5.5	4+1 model	50
5.5.1	Development view	51
5.5.2	Logical view	52
5.5.3	Process view	54
5.5.4	Physical view	56
5.5.5	Scenarios	56
6	Development	59
6.1	Blackboard development	59
6.1.1	Development support	59
6.1.2	Implementation strategies	61
6.1.3	The Blackboard API	62
6.2	Prototype	64

6.2.1	Building block creation	64
6.2.2	Front-end	67
6.2.3	Back-end	73
6.2.4	Logic	76
6.3	Testing	77
6.3.1	Unit testing	77
6.3.2	Integration and system testing	78
6.3.3	NTNU test server	78
7	Developer experience	81
7.1	Prerequisites	81
7.2	Blackboard as a development platform	82
7.2.1	Full integration advantages	82
7.2.2	Platform limitations	83
7.2.3	Deployment and testing	86
7.2.4	Blackboard support	86
7.3	Challenges	88
8	Results and discussion	91
8.1	Questions	91
8.2	Contributions	93
8.2.1	Module development	93
8.2.2	Extended functionality	94
8.2.3	Blackboard Summary	95
8.3	Project evaluation	96
8.4	Future Work	97
8.4.1	Student data gathering	97
8.4.2	Refined and extended functionality	97
8.4.3	User interface	98
8.4.4	Extensive testing	98
9	Conclusion	99
	Bibliography	101
	Appendices	107
A	Peer review module pages	109
B	Unit and integration tests	115

List of Figures

1.1	Illustration of the research process [7]	4
2.1	An illustration of a simple peer review process	7
2.2	Behavioral, cognitive, and constructivist perspectives	11
2.3	Relevant aspects of a pedagogical strategy using peer review	12
2.4	ADDIE instructional design model [32]	13
3.1	Example of bad affordance - inside the red box is a link text	18
3.2	Student peer review in Blackboard [48]	19
3.3	Student peer review in Canvas [50]	20
3.4	Rubric self evaluation example [52]	21
3.5	ItsLearning subject overview	21
3.6	PeerMark peer review example [55]	22
3.7	Calibrating peer review example in CPR [58]	24
4.1	Flashcard example - StudyBlue Android application [60]	31
5.1	Illustration of the peer review process using the peer review module	34
5.2	Blackboard bbNG tags versus HTML tags	37
5.3	A standard blackboard page	38
5.4	Assessment creation process page overview, from the current Blackboard peer assessment module	39
5.5	Early sketch of evaluation view(as seen by the evaluator)	41
5.6	Illustration of a list over student submissions with evaluation scores ready to be given the final score	42
5.7	Illustration of evaluation numbers option and anonymity option that would be added into the assessment creation form	42
5.8	Layer diagram of the peer assessment module	51
5.9	UML Class Diagram	53
5.10	BPMN: Process view of the peer assessment creation	55
5.11	An illustration of the peer assessment module topology	56

5.12	Use Case: Create peer assessment	57
5.13	Use Case: Submit assessment answer	57
6.1	Part of the REST API for interacting with Blackboard courses	62
6.2	Peer review module code showing the RequestMapping method that creates the ModelAndView object for create.jsp	63
6.3	System admin page in Blackboard virtual machine environment	65
6.4	Code snippet required to enable use of HttpServlet in JSP files	67
6.5	Code snippet from the student_reviews.jsp used in the peer review module	68
6.6	Code snippet from the create.jsp - function for deleting a criteria scale	69
6.7	An overview displaying the different JSP pages used in the module and the connection between them (High Resolution Image, zoom is possible)	70
6.8	A list over student submissions with evaluation scores ready to be given the final score	71
6.9	The final version of the assessment evaluation view (as seen by the evaluator)	72
6.10	MVC spring architecture [78]	73
6.11	Assessment Controller for the peer review module (Pruned to better visualize the Request Mapping Methods)	74
6.12	ER diagram over the peer review module database entities	76
6.13	Peer assessment successfully created on NTNU test server	79
7.1	Code snippet showing how our software accessed users and their information	83
7.2	The different versions of Blackboard, where the Ultra Course View enables a more dynamic user interface. Currently NTNU uses the Manage Hosted 9.1 version	84
7.3	Code snippet showing how the peer review module sets the students grade	85
7.4	The OSCELOT project page containing a large amount of Blackboard projects [69]	88
7.5	The correlation between the content-handler tag inside the manifest, and the appearance of a pressable link in Blackboard	89
8.1	Part of the review of review page, displaying a student giving feedback on an evaluation	94
8.2	Blackboard student mobile application [80]	98
A.1	Pressing the peer assessment option in course content	109
A.2	The assessment "Write an initial research proposal" as shown inside the course content list	110
A.3	The assessment "Write an initial research proposal" as shown inside the course content list	111
A.4	The submission page	112
A.5	The students review page, showing all submissions that the student should evaluate	113
A.6	Part of the review of review page, which shows a student giving feedback on an evaluation	113

Abbreviations

LMS	=	Learning Management System
API	=	Application Programming Interface
JSP	=	Java Server Pages
ID	=	Instructional Design
ISD	=	Instructional System Design
UI	=	User Interface
BPMN	=	Business Process Modeling
UX	=	User Experience
FR	=	Functional Requirement
MVC	=	Model View Controller
UML	=	Unified Modeling Language
URI	=	Uniform Resource Identifier
DAO	=	Data access object
REST	=	Representational state transfer
SOAP	=	Simple Object Access Protocol
LTI	=	Learning Tools Interoperability
STS	=	Spring Tool Suite
VM	=	Virtual Machine
IDE	=	Integrated Development Environment
ER	=	Entity relationship

Chapter 1

Introduction

Utilizing assessments is a fundamental part in current education. Most assessments are just created by the instructor and delivered by the students for the instructor to evaluate. But what if the students could become the key factor in a larger part of the assessment process? Peer review is an assignment methodology with this as one its focal points, focusing on students evaluating each other. It has existed for a long time, and is widely used in academic writing, but has lately become more and more relevant in an educational context as well.

Peer reviewing encourages the students to take on the role of the instructor. And thus, by including them more actively in the evaluation process, they are more likely to have an even higher learning outcome than with standard submission assignments.

In order to utilize peer reviewing properly it has to be easily available and uncomplicated to use. One way to achieve this is to digitize peer review by integrating it with the Learning Management System (LMS) in universities, making it immediately accessible for both instructors and students. The possibilities of LMS integration are nearly endless, hence creating tailored LMS content by digitizing assessment methods is becoming increasingly relevant.

1.1 Background and Motivation

Many positive aspects has been discovered from the use of peer reviewing in an university context. Earlier conducted research indicates that students participating in peer reviews achieves a more thorough learning and understanding of the subjects [1]. Also, the students improves on their reflection, analyzing and assessing skills by reading and evaluating the submissions. Toppings article on peer assessment mentions how the method can complement other approaches, such as cooperative learning. And that the results of using peer assessments can help learners identify both their strength and weaknesses [2].

Another positive aspect of making use of such an assignment form is that the instructors does not need to use too much of their time evaluating every submission, as they are most likely evaluated thoroughly enough by the student peers, especially in the case where there are more than one evaluation per submission. As a result, the instructors are able to use their time on other course specific aspects [3].

A motivational factor for selecting peer reviewing is the digitization process that has happened in the later years. Earlier, when one had to do peer reviewing the analog way, it could potentially cause a lot of work for both the teacher and students (having to print assignments or manually assign papers to students). But now, with a digital platform the distribution of papers can be done automatically, and the reviewing itself can take place outside of the classroom increasing the flexibility for the instructors and students [3, 4].

This opens up a lot of opportunities, like creating fully digitized E-Learning platforms, making it possible for students all over the world to attend. An example of this can be taken from Sweden and the article *E-Learning: The Digitalization of Swedish Higher Education*. There, they created an E-Learning platform called Net University, which was a consortium of 35 different colleges in Sweden [5].

The Blackboard Learning Management System [6] was to be introduced at the Norwegian University of Science and Technology (NTNU), hence making the digitized content inside it available for all instructors and students at NTNU. This provided an unique opportunity as the LMS provided an open development platform for integrating digital content, and could therefore be used to digitize tasks and assessment methodologies. And with courses like Web Technology at NTNU being interested in utilizing a digitized version of peer review, this was something that could be made possible through Blackboard.

Blackboard has long been an open platform where developers could integrate their own custom modules. But despite the fact that parts of the platform is open, a small amount of it is still closed for the public. This raises questions about which bottlenecks and development issues NTNU would encounter, if students or employees wished to develop their own Blackboard modules. Finding answers to these questions could potentially enhance the process for others who were working on or considering Blackboard development.

This led us to looking at the possibility of digitizing peer reviewing, and ways to implement and integrate such a module into the Blackboard Learning Management System. And hence our research questions aims to find out how peer reviewing works, identify what important features a digital implementation of peer reviewing would require, and map potential bottlenecks in Blackboard development.

- RQ1:** How is the peer assessment method currently utilized in an educational setting?
- RQ2:** Which features are important to include when implementing a digital peer review module?
- RQ3:** What potential bottlenecks and concerns needs to be considered when developing a Blackboard module?

1.2 Project Details

The main purpose of this section is to give an understanding of the project main goals as well as the chosen strategy trying to solve these in the best possible manner.

1.2.1 Description

The objective of this project was to investigate the creation of a digital peer review module through the use of Blackboard as a development platform. This development aimed to reveal bottlenecks and concerns, as well as providing information on developer experience for future developers.

A design of a peer review module was made, before it was implemented and integrated into Blackboard. In the design, it was essential to gather qualitative information required for the functionality to be specified, as well as potentially identifying other sought modules.

The development of the peer review module was through the use of frameworks such as Spring and APIs offered by Blackboard. It was implemented using Java for back-end logic, with JavaServer Pages (JSP) and JavaScript for front-end. This is described further and in more detail in Chapter 6 - Development.

1.2.2 Approach

To start off, a literature review was conducted to provide a conceptual framework as a foundation for the rest of the project. This foundation was then used to make the research questions as precise and well formulated as possible. The research questions were focusing on different aspects of digitizing the peer assessment method in a university context. And as this project followed an experimental development paradigm, it was also important to examine the possibilities of using Blackboard as a development platform.

To be able to answer the questions in a good manner, it was necessary to conduct several semi-structured interviews to gather qualitative data for the design and development phases. The research process as a whole can be seen in Figure 1.1.

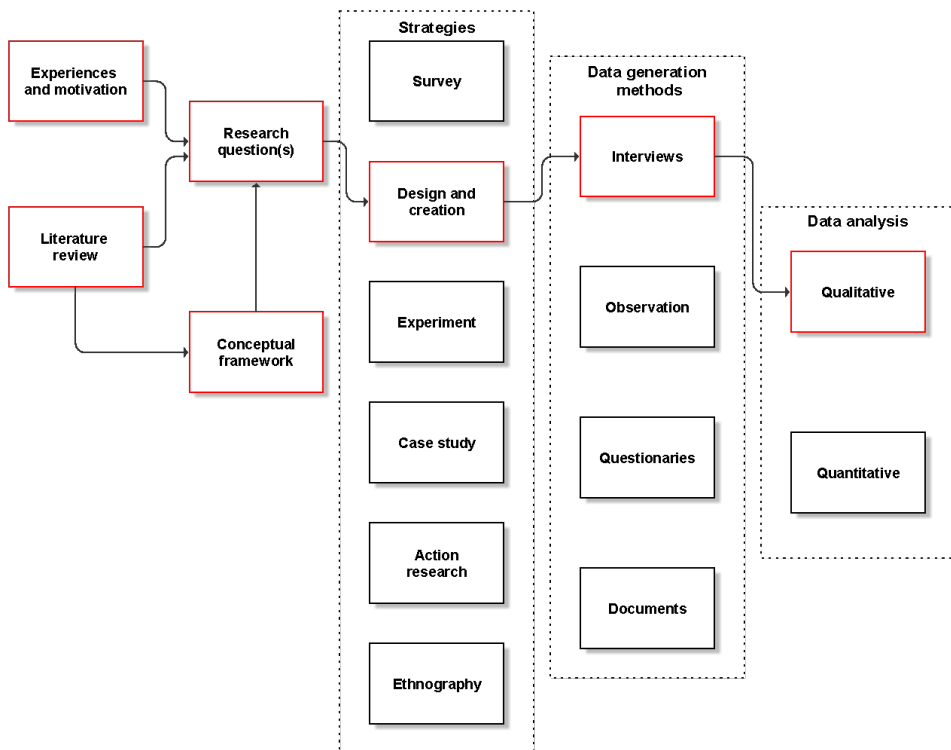


Figure 1.1: Illustration of the research process [7]

After gathering the data from the interviews, it was important to structure the information in a good way for it to be possible to identify and specify the different requirements for the design and development of the digital Blackboard module.

Designing and developing a fairly complex module, required a good amount of integration into already existing functionality in Blackboard. This resulted in many different experiences to give an indication on the possibilities and limitations of Blackboard as a development platform. A detailed summary of these experiences is elaborated in Chapter 7 - Developer experience.

1.3 Thesis Outline

In this section, a list of all the different chapters in this thesis as well as a short description of each of these is presented. This, to give an overview of the thesis as a whole.

- **Introduction**

This chapter is meant to give an overview of the motivation, background and approach used for this master project.

- **Background**

The background describes much of the work that was conducted before the actual design and implementation phases. This includes detailed theory about relevant topics such as peer review, pedagogy, and motivation.

- **State of the art**

In this chapter, the state of the art on relevant technologies are presented, more specifically peer review modules, both integrated into Learning Management Systems and third party modules.

- **Interviews**

There was conducted four semi-structured interviews with different professors at NTNU that had some experience with peer review systems. These interviews are presented in this chapter, and were used to identify important features and functionality for the design and implementation phases.

- **Design**

The design chapter presents the work conducted when preparing for the development phase. This includes a list of stakeholders and target users, a description and design of the chosen interface, and the software's functional and quality requirements. The chapter also contains Philippe Kruchens 4+1 model, which describes the different parts of our module seen from different views.

- **Development**

Here, it is described how to start developing digital modules for the Blackboard LMS, and what development alternatives that are available for them. After that, the prototype phase containing the choices made in this project and illustrations of the final product is shown. Lastly, our testing phase is explained.

- **Developer Experience**

A lot of information and experiences was gathered during our development process, and the developer experience chapter aims to help the future Blackboard developer by highlighting some of it. This includes prerequisites, our experience with Blackboard as a development platform, and challenges met during the development process.

- **Results**

The results chapter presents a summary of the answers on the research questions, the main contributions of this thesis, a self-reflective evaluation of the project as a whole as well as future work that could be included to improve the module even more.

- **Conclusion**

Our final impressions of the project, as well as the final judgment on how well the research achieved its goal.

Background

This chapter contains the theoretical research that was the foundation of this master thesis. The key theoretical aspects is peer reviewing, pedagogical theory and motivational theory, all of which are connected to each other.

The background focuses on the importance of gathering knowledge about how relevant aspects work. As this project was based on designing and developing a digital peer reviewing module for Blackboard, the most important concept to understand was peer reviewing as a task type, especially in a university context.

2.1 Peer review

There are a big range of assessment methods available in today’s education, ranging from formative assessments with formal or informal techniques, to summative assessments like exams or final presentations [8]. Peer review, also called peer assessment, is a method where the students are to assess each other based on criteria set by the teacher or professor. The students has to review and understand their peer’s work and give meaningful feedback as a response (Figure 2.1). This, to enhance the learning outcome for the students, as well as giving the professors more time and resources to focus on other parts of the course, such as the lectures and the exercises [9, 10]. This section digs deeper into the understanding of peer review and the requirements for developing a successful peer review module.

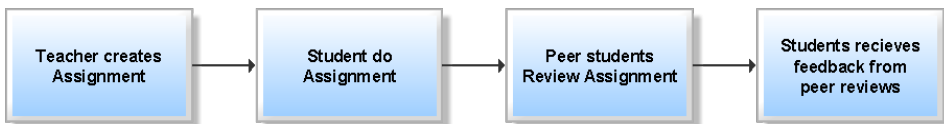


Figure 2.1: An illustration of a simple peer review process

2.1.1 Peer review in education

Peer reviewing is not a new concept, as it has been used for a long time in quality assurance of research papers and articles in the field of academic research [11]. There, publishers of research articles send in their manuscript to other scientists from the same field of study for evaluation. The scientists give a feedback containing discovered errors and weaknesses as well as a recommendation stating if the manuscript should be published or not.

The peer reviewing discussed in this paper is not connected to research papers, but rather on allowing students to participate in the assessment of their fellow classmates. This is something that only in the recent years has become more and more utilized in an educational context.

Studies conducted by Falchikov shows that peer assessment deepens students understanding of their own learning and empowers students to become more actively engaged and self-directed in their learning processes [1]. Students also gain skills in writing, describing, assessing, reviewing and analyzing, when challenged with the task of creating good and thorough feedback on others assignments. In addition, peer reviewing can provide benefits for the instructor, if the instructor creates non-graded exercises where the students are responsible for reviewing each other the teacher reduces their work load [3].

In the later years a lot of studies has been conducted on the pedagogy behind peer review, and these indicate that peer reviewing can potentially provide the learners with the opportunity to develop a wide range of skills [12]. This includes being able to aid their fellow students, giving them skills in collaborative learning and instructional scaffolding. It can also increase the learners capacity for independent problem solving and reduce the learners dependence on teachers [13] [14].

In spite of this, not all results on peer reviewing are positive, and there are studies that showed little to no improvement when using peer reviewing [15]. In addition to some studies with low improvement rate, there were studies where the students did not prefer peer reviewing. In a study conducted by Hu and Lam, the majority of the learners responded that they prefer a teachers response over a peer review. Mostly because they felt that the teachers response was more accurate and as they did not feel they could trust that a fellow students answer was as correct as the teachers answer. However, it is important to notice that the study showed improvement on student performance from the peer reviewing [12].

As mentioned, several reasons has been added to why peer reviews has gained such mixed results. Some of the studies highlights the issue that students participating in the peer reviewing are still in the process of learning the subject themselves and therefore they do not feel experienced enough to question other students work [13]. It is also an issue that the students participating are not familiar with peer reviewing, thus they are not equipped to offer useful feedback as evaluators [16]. And hence, it is essential that the instructor prepares the students before engaging them in peer review assessments.

2.1.2 Peer review practises

An important aspect of using peer reviewing for the first time is to introduce the idea and approach of doing a peer review before giving the students their first assignment. The introduction can be done by explaining the importance of revision, going through evaluation criteria, or simply running a practice session where students get to test peer reviewing on a smaller scale. The purpose of this is to make the student understand the goal of doing peer reviewing, thus motivating them to do the reviewing properly and thorough. It is also a good idea to instruct the students in both giving and making use of constructive feedback, as many students may have had little previous experience in giving written feedback to others [3, 2].

When implementing peer review it is also important to consider the degree which you want the students to be involved. For example, involving students in the creation of guidelines or evaluation criteria for rectification could increase accuracy when students do the peer assessment. And hence, at the lowest level of involvement the instructor creates and prepares all guidelines and evaluation criteria, while answers and all sorts of scores given by students are recommendations only. While at the highest level, the students work together with the instructor to write guidelines and criteria for assessment and scoring [17].

Guidelines, evaluation criteria and scoring can have a large impact on how well the students perform their peer review [18, 19]. This means that it is important that the educator selects some guidelines to discuss with the students, and sets a fair scoring or grading form. Guidelines for peer review can be as simple as "read the whole text before reviewing", or more complex and focus on reviewer behavior like "offer suggestions not commands" or "do not let your own opinion bias the review" [3].

Scoring and grading must also be carefully considered as it can cause student behavior to alter if there is grading involved [20]. It is worth noting that at the Norwegian University of Science and Technology, the students has to be graded by a teacher, so in the case of a peer review the teacher would need to do a final grading of the assignment. Scoring, as well as grading, can affect student performance and many of the teachers that were interviewed for this paper agreed that scoring can be an extrinsic motivation [Section 2.3.2], but that it should always be approved by an educator or teaching assistant [21] [Chapter 4].

2.1.3 Effective implementation and possible limitations

In addition to following the implementation guidelines, there are some recommended rules the educator should follow in order to gain maximum efficiency from the peer review [3]. This includes:

- Making the guidelines for the students easy to understand and make sure that the students are aware of their importance and usage.
- Ensure that the focus of the students reviewing is on useful feedback and not on the grading.

- Monitoring the students usage of the guidelines and provide ways for them to learn how to give constructive and descriptive feedback.
- Highlighting the benefits of being a peer assessor to the students, empowering the students to become more self-directed learners.

There are also some limitations when introducing peer reviewing to students. First is the question of student knowledge versus the teacher's in reviewing the assessments, as many students feel that the teacher would provide a more knowledgeable answer. This, as the peer review is meant to complement teacher feedback rather than preclude it [22]. Secondly, there is the issue of bias in student reviews. While many students will try their best there will be some who just wants to get it done as quickly as possible. The learning outcome from a quick review is of course very little, thus both educators and the peer review software in question needs to highlight the difference between a good and a bad review. Thirdly is the problem of collaboration between student groups, as some students may plan and vote on each other in order to get a good score. This is of course more difficult if a written review is to be provided as well, but it means that the educator and software needs to be aware of this and be able to prevent such actions. Research conducted by Lu and Bol indicates that making use of anonymous peer reviews would result in more thorough and critical evaluations, which also potentially could solve the problem of student groups voting on each other as they would no longer know who they evaluated, at least to a certain extent [23]. Lastly is the question of how much time peer review can save the instructor. While it initially can seem like the students are doing all the work, the time spent planning and initializing peer review must not be underestimated. Falchikov has cautioned that in the short and medium term the instructor will not save much time on the use of peer assessment. The reason for this is because of the effort and time it takes to establish high quality peer assessments [24].

2.2 Peer review in a pedagogical setting

In order to understand peer review even better, it was important to research on the overall concept containing and making use of peer review, namely pedagogy. Pedagogy is the method and practice of teaching, especially as an academic subject or theoretical concept. There are many existing pedagogical aspects, and in this section, some of the most central and relevant ones are presented, starting with describing pedagogy in a general fashion.

2.2.1 Theoretical aspects

Pedagogy concerns the study of how best to teach and specify instructive strategies. Learning how to manage activities and instructions in the classroom can give educators the power to optimize the learning potential of every student. Therefore it is crucial that the educators understands pedagogical theories, instructional theories and has a pedagogical strategy for their education [25].

There exists a range of different theories about pedagogy. These often base themselves on learning theories [26]. Pedagogical theories are prescriptive (tell how people should

learn) while the learning theories are concerned with how people acquire, retain and recall knowledge. There are many learning theories, but three of these, namely Behaviorism, Cognitivism and Constructivism (Figure 2.2), are considered central. These theories defines; how the learner should process given stimuli, when knowledge is considered "transferred", and who has the most responsibility in the learning process (teacher or student) [27]. And while these learning theories focuses on the learning process they often also address pedagogical issues. It is crucial to understand that no single learning theory can address the requirements of every teaching and learning event. Thus a combination of perspectives is required in order to achieve an efficient learning approach [28].

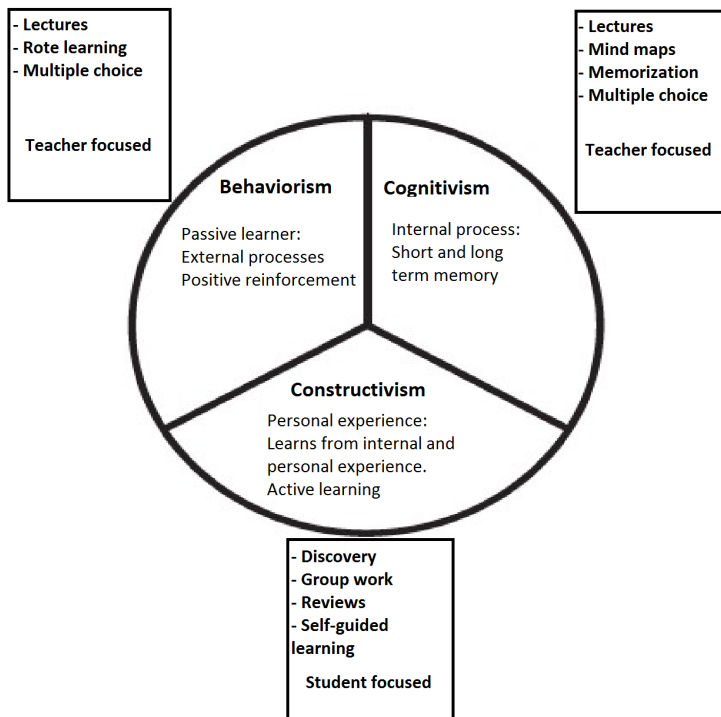


Figure 2.2: Behavioral, cognitive, and constructivist perspectives

Pedagogical theories describes how to teach and make people learn different topics. And to be able to make university students learn through the use of a peer review module in a digital learning system it is important to identify and highlight the theory behind it. To know when to make use of peer review in an educational context, it is decisive to understand the concept of a pedagogical strategy and how the choice of such an assessment method is relevant to this.

2.2.2 Peer review as part of a pedagogical strategy

Pedagogical strategies consists of multiple different theories and design approaches, and the process of building a pedagogical strategy is the process of selecting the appropriate theories, pedagogical approaches, and designs. An illustration of which relevant aspects that are part of a Pedagogical Strategy revolving around peer reviewing is given in Figure 2.3.

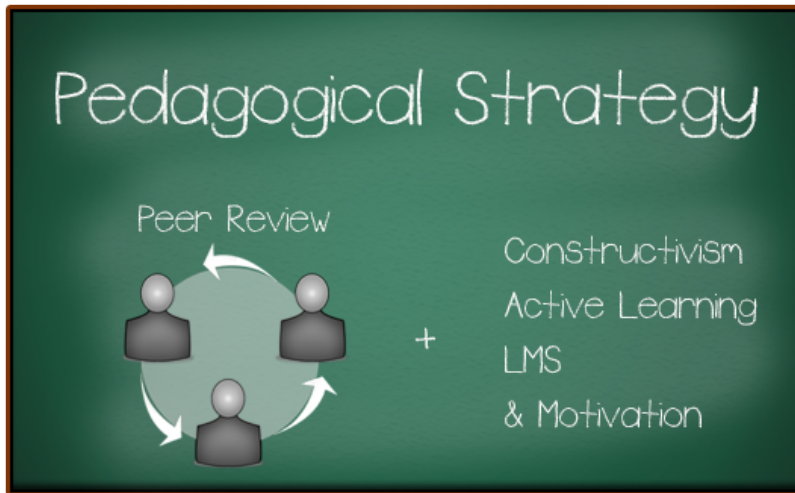


Figure 2.3: Relevant aspects of a pedagogical strategy using peer review

Whilst the underlying theories of learning remain largely consistent, pedagogical approaches are constantly evolving. There are many different approaches and their goal is to provide a teaching approach according to need, audience, and topic. An example of a pedagogical approach is active learning where students engage in activities such as reading, writing, and evaluation of class content. Active learning strives to more directly involve students in the learning process. Simulation, use cases and problem-based learning are some directions that promotes active learning [29]. The use of peer review as an assessment method can be seen as part of the pedagogical approach, and in the case of active learning, the choice of such an assessment method could be wise, as it strives to engage the student to learn through the use of an active evaluation approach to support self-regulated learning [30].

Choosing an appropriate pedagogical approach can be the key to getting the students engaged in the course content, but it often requires years of experience as an educator [28]. Finding pedagogical theories and approaches are only the first part of the pedagogical strategy, selecting a design strategy is the next step.

Instruction theories covers how you should design the learning process in a way that is both motivational and at the same time fulfills the educational purpose. Instructional design (ID), and instructional system design (ISD) are practices that are focusing on creating and

strengthen the learning process by making it appealing and effective for learners to acquire and learn new knowledge [31]. It has been developed over a 100 different ISDs throughout the years, but most of these are somewhat based on the more generic ISD called the ADDIE model. This model defines some steps that has to be conducted to create a system that enhances and facilitates learning from a general perspective (Figure 2.4).



Figure 2.4: ADDIE instructional design model [32]

Once an appropriate approach is chosen and a design strategy is selected, the final step is directed towards educational technology [33]. This is a wide field consisting of technologies used in education with the assumption that it can facilitate pedagogical scenarios. In this thesis, the focus is on Learning Management Systems, a technology mainly focused around knowledge transfer, but which also includes tutoring and coaching.

2.2.3 Educational support in LMS

Learning Management Systems gives the professors and students the opportunity to take the learning and course content to an online platform. LMS can provide tools like; forums for discussion, group forming tools, video/audio communication and creation of different assignments, such as peer reviewing. Blackboard [6] for example, offers the whiteboard feature, which enables instructor to get all students to come online in order to explain spe-

cific points [34]. While Moodle [35] enables learners to use scales to rate or grade forums, assignments, quizzes and more. All these tools are often spread across several different LMS, but each system offers different pedagogical tools that the course professor can use.

According to research conducted by Hinostrozaa and Mellar, when developing pedagogical software, to for instance LMS, one has to take into account the browsing strategy of the user, and not only the learning strategy. The browsing strategy focuses on the way the user is browsing content, making it important not to only publish and display as much content as possible, but also organize it in a meaningful and pedagogical matter [36]. Thus, this was something that needed to be taken into account, regarding the complexity and information display, when developing a peer review module.

LMS also has some pedagogical issues, mainly because teachers and faculty members often use such systems only to "put content online" without applying any pedagogical principles. The issue is also connected to the lack of pedagogical affordances of existing LMS and many criticize these for not supporting more modern learning theories like constructivist learning [37, 38].

2.3 Motivation

When focusing on the student perspective of our module it was important that the software was motivational to use. In order to understand this, learning theories that gave insights in how motivation was connected to and utilized in peer reviewing was essential. In this section we look at motivation and motivating factors from both a general and student perspective.

2.3.1 Motivation - an essential aspect

Motivation is probably the most important factor for educators who wants to enhance learning [39]. In order to motivate, one must understand what motivation is and identify elements that are motivating for a student.

Motivation is defined as the act or process of motivating; the condition of being motivating; a motivating force, stimulus, or influence; incentive; drive; something (such as a need or desire) that causes a person or student to act; and the expenditure of effort to accomplish results [39]. There is no single theory on motivation that explains all human motivation, because human beings are complex creatures with complex needs and desires.

Student motivation is an essential element that is necessary for quality education. When students are motivated they; pay attention, start working on tasks immediately, asks questions, and appear to be more happy and eager [39]. All these attributes are usually essential, regardless the type of activity. Thus, while developing software modules based on such activities, like peer reviewing, it is often very important to have in mind how to achieve these attributes in a good manner.

2.3.2 Peer review - Intrinsic or extrinsic motivation?

Intrinsic motivation is characterized by a focus on the inherent satisfaction in performing a particular behavior for its own sake, in contrast with extrinsic motivation, in which the focus is on attaining some separable outcome [40]. Behavioral research suggests that a sense of autonomy, or being in control of one's choices, facilitates intrinsic motivation [40].

For a student, the intrinsic motivation would be involvement (the desire to be involved), curiosity (find out more about their interests), challenge (figuring out the complexity of a topic), and social interaction (creating social bonds). Extrinsic motivational factors include compliance (to meet another's expectation, to do what one is told); recognition (to be publicly acknowledged); competition; and work avoidance (avoid more work than necessary). Intrinsic motivation has emerged as the most important one for motivating the students as it results in higher quality learning and creativity. Even so, teachers cannot always rely on intrinsic motivation and must therefore understand the consequences of the extrinsic alternatives [41].

Research conducted by Tseng and Tsai on Taiwan college students gave indications on that students who had been engaged in peer assessments were highly motivated. And from the questionnaire results, it showed that the students did participate in these activities for enjoyment or challenge, thus the intrinsic motivation was the dominating motivational factor [42]. In spite of this, if any assessment type utilizes scores that affects the grades, it can be seen as an example of targeting the compliance of the students. And hence, peer reviewing making use of scores to motivate the students, can be seen as both an intrinsic and extrinsic assessment method.

2.3.3 Goals as guiding criteria

It is important for students to have a goal, whether it is learning goals, goals for a single lesson or goals tied to performance or evaluation. [39] Ford defined goals as the desired end state people try to attain [43], and goal-setting theory on motivation states that specific and challenging goals along with appropriate feedback contribute to higher task performance [44]. Goals help the student focus towards completing a task, and gives them a clear direction. Lock found evidence that there is a relation between a goal's difficulty and task performance, which more recent studies supports as well [44]. Studies also showed that specific challenging goals let to higher output than vague goals like "do your best"[44]. This can be applied to peer assessments, as it is wise to create criteria that are specific with an appropriate difficulty, to guide the students in the right direction. This can both be utilized in the guidance submission delivery, but more importantly when the students are to evaluate each other.

2.3.4 Social effects

Both in psychology and in teaching, motivation is discussed as an individual experience. Teachers often talk about students that lack motivation because of influences beyond the classroom environment. Research done by Urdan and Schoenfelder shows that this view

is inaccurate and that seemingly unmotivated students can become willing participants in tasks if the tasks are tailored to their interests, or if it gives them the opportunity to fulfill social needs like working with friends [45]. This also puts some responsibility on the teacher who has to create a learning environment that supports student motivation (e.g. an environment where the student is supported, socially connected with the teacher and peers and where they are allowed to take ownership for their learning).

Student engagement in learning has also been shown to increase if the tasks provides some sort of social interaction with other students or the teacher [39]. This can be related to peer reviewing, as one of the perks of this as an assessment method is that the students gets connected together, even if only implicitly through the use of anonymous evaluation.

Chapter 3

State of the art

A good system is a system that has learned from others success and mistakes and reused previously discovered knowledge in a new and intuitive way. In order to make a good system one must have an overview of functionality used in similar systems, and at the same time understand which of them actually works. This chapter does not only identify former known functionality by looking at state of the art software, it also presents the amount of dedication developers put into peer review modules used within the learning process. For our research it was important to identify what solutions and functionality that was already out there, not only for the progress of new functionality, but in order to ensure that we built upon previous knowledge and not just remake something that already existed.

This section is divided into peer review modules integrated into Learning Management Systems and peer review modules developed as third party software.

3.1 Peer review in Learning Management Systems

Learning Management Systems are powerful environments. They do not only focus on the knowledge communication between students and professors, but also aspects such as administration, delivery and evaluation, tracking and curriculum management [46]. Having a LMS that fosters motivation for the end-users requires a high usability and effectiveness.

There exists a range of different Learning Management Systems out there. In this section, a chosen subset of these are presented with their corresponding peer review module.

3.1.1 Blackboard

Blackboard [6] is a LMS that at the time this report was written, was about to replace It-learning [47] at NTNU. Blackboards main focus is on student-professor communication, but in addition to this it also has a much more open and easy way to develop and test digital modules compared to many other LMS. By using the Blackboard API anyone can

develop Blackboard modules, thus opening the gate for outside developers who wish to add functionality to the existing LMS structure.

Blackboard does not contain as much excess functionality as for example itsLearning, but there are still some UI aspects that needs to be worked on in order to increasing the usability of the system. Even so, Blackboard has great potential in enhancing the intrinsic motivation of the students.

When it comes to peer reviewing, Blackboard already has an integrated self and peer review module (Figure 3.2). In it, the teachers can create questions and answers, as well as add criteria to each question that should be evaluated based on. The self and peer review assessment is separated from the standard assignment module, as it is a bit different. It is positive that Blackboard already comes fitted with a peer review option, but the option is unfortunately not without faults. It is not possible to import assignments created with the assignment module into the peer review module, which can be counter intuitive for the users. The module itself is also very complex as it contains a lot of different views and configurations. Most of the complexity revolves around creating the assessment, as it is much more simplistic for the students when they are to deliver the exercises and evaluate others. Apart from complexity, the affordance of the module is also very low, an example of this is shown in Figure 3.1 where the question-links has no indication of being clickable. Overall, the peer review possibilities of Blackboard are fine, but still has a strong potential for improvement.

Ta evaluering: Test Evaluering

Evalueringsnavn	Test Evaluering
Instruksjoner	Test Evaluering
Evalueringer å fullføre	Evalueringer fra andre: 2 Egenevaluer din egen innsendte evaluering
Innsending	1. mars 2017 12:31:00 til 8. mars 2017 12:31:00
Evaluering	8. mars 2017 12:31:00 til 15. mars 2017 12:31:00

Spørsmål 1

Spørsmål

Status: Ikke fullført

Figure 3.1: Example of bad affordance - inside the red box is a link text

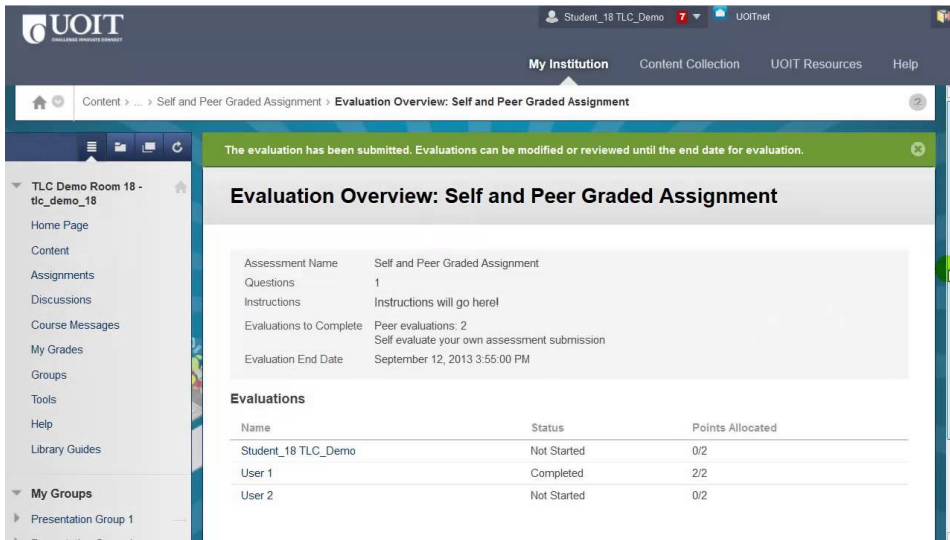


Figure 3.2: Student peer review in Blackboard [48]

3.1.2 Canvas

Canvas [49] as a LMS can be described as very clean and modern. The design as well as the choices made regarding features of the different modules shows that they value usability over complexity and a lot of functionality.

To create a peer review assessment in Canvas the teacher has to create a normal assignment first. Then they can add a peer review assessment to the given assignment. The distribution of submission evaluations can be done manually by the professors or automatically by Canvas. It is also possible to add peer review assessments to group assignments, but not to assign peer reviews to entire groups. The peer assessment itself can be very simplistic, as the students may only need to comment on other students assignments, but it can also be more complex, through the use of rubrics created by the professor for students to evaluate and give score based on (Figure 3.3). Even so, the professor is the one that has to give the final evaluation and grade to the assignment.

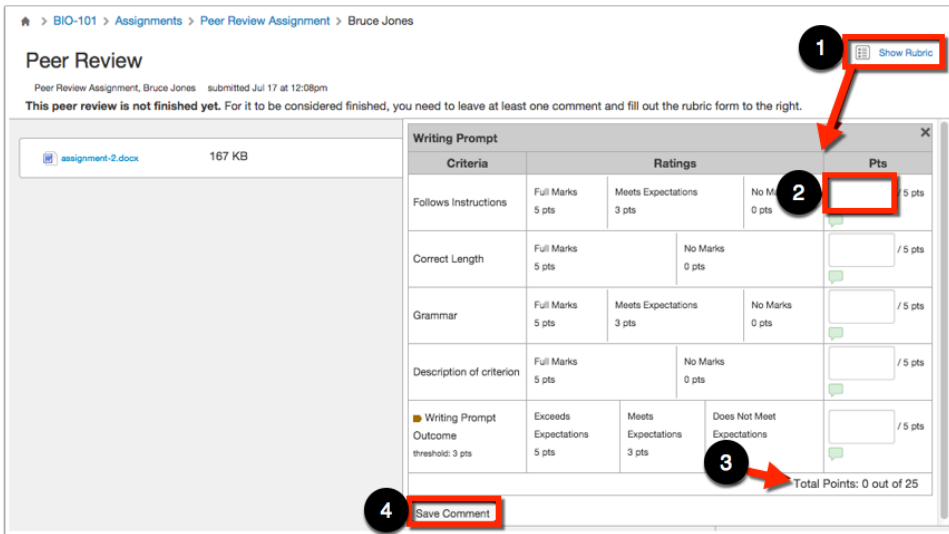


Figure 3.3: Student peer review in Canvas [50]

3.1.3 Google Classroom

Google Classroom [51] is a LMS that are very heavily focused on humanities subjects and not so much on subjects such as mathematics and science. This, because the LMS is based on the functionality that are available on standard Google Drive, with modules focusing on writing textual documents of different kinds. For the design, Google Classroom is very clean and streamlined, which makes everything in it intuitive and easy to use.

Google Classroom can make the use of rubric templates to peer evaluate the students. To do this, one need to import all the student submissions into a Google spreadsheet, creating a link that redirects to an evaluation page for each submission. To see who evaluates who, the students has to write their names in a column related to the row of the assignment.

A rubric template has the possibility to specify scores, criteria and the weight of these. A downside is that one has to specify all of these for the rubric to be valid. An example of a rubric template can be seen in Figure 3.4.

To conclude, the Google Classroom peer review software is very simplistic, but also a bit cumbersome to create and make use of. The fact that these peer evaluations only works on assessments given in the formats specified on Google Classroom (and Google Drive), as well as the small amount of customizability of the rubrics, makes it hard to tailor for different kinds of task types and courses.

3.1 Peer review in Learning Management Systems

Teacher Score		Teacher Note	Student Justification	Student Score	Criteria Weight	Rubric Criteria	Advanced	Proficient	Basic	In Progress	Unacceptable
			I struggled with finding a good model. I think I need more details to make it more authentic.	3		Is a good model of mathematics in the real world. Could use some improvement.	100%	85%	75%	65%	0%
			I applied the algorithms but did not justify my thinking	2	30.0%	DOK	DOK 4	DOK 3	DOK 2	DOK 1	DOK 0
			I showed my work and believe the calculations are accurate.	4	15.00%	Math Calculations	Accurate math calculations	Minor calculation mistake	Several calculation errors	Major calculation errors	Math is completely incorrect
			I correctly demonstrated parabolas at at DOK 2	4	20.00%	Parabolas	Model demonstrates understanding of parabolas	Model demonstrates understanding of parabolas. More detail needs to be included	Model addresses parabolas. Full understanding of concept not coming across.	Model addresses parabolas in a minor way	Model does not address parabolas

Figure 3.4: Rubric self evaluation example [52]

3.1.4 ItsLearning

ItsLearning [47] is mainly focusing on the student-professor communication, and not how to create knowledge between students. The core functionality on itsLearning tries to make it as easy and orderly as possible for the course responsible to create informational announcements and add/post different kinds of curriculum relevant files (Figure 3.5). Unfortunately, itsLearning is very strict when it comes to developing plug-ins for extending and adding functionality, thus making the development of modules a cumbersome process. It also contains a lot of excess functionality that never gets used, hence lowering the usability as well as increasing the time it takes for a student to learn and get to know the essentials.

ItsLearning does not have a peer review assessment module integrated, but as it is a LMS that is used by many different institutions in Norway, it was important to take into account.

Figure 3.5: ItsLearning subject overview

3.2 Third party peer review software

There are also peer review software that are created as third party software, which can be integrated into for instance Learning Management Systems or other web applications. The advantage of these systems is that they can be used in multiple different LMS without having to be directly integrated. The disadvantage however, is often the lack of integrated support with the LMS, like direct grading or storing results internally.

3.2.1 PeerMark

PeerMark [53] is a peer review tool created by the Turnitin group [54]. This module is used for extending the normal assignment module in Turnitin. After the instructor has created an assignment, they can create a PeerMark assignment that starts after the normal assignment has ended. The PeerMark assignment has a range of different options that can be selected. One can for instance specify if the reviews are to be anonymous or that students that have not delivered a normal assignment can still do a review. It is possible to specify that the software should automatically assign the reviews between the students, but the instructors can also add other rules, stating that certain students are not allowed to do the assignment or manually override the pairing of specific students.

After creating the PeerMark assignment the instructor can create two kinds of criteria, free questions or scale questions, both displayed on the right side in Figure 3.6. The former are questions the evaluators must answer during their review and write a minimum amount of words, while the scaling questions are where the evaluators has to answer on a scale from for instance 1 to 5 (defined by the instructor). This was just a subset of options in PeerMark, as there are other less important functionality available. All in all, this is a very clean in orderly peer review module.

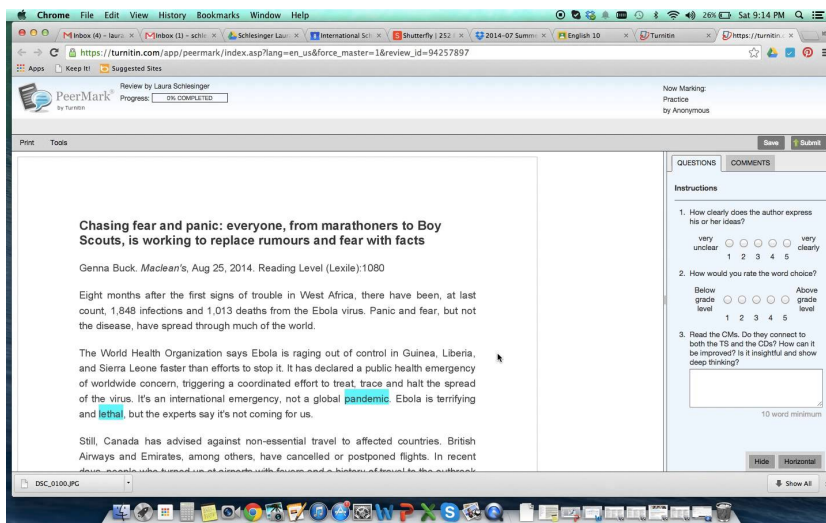


Figure 3.6: PeerMark peer review example [55]

3.2.2 PRAZE

PRAZE [56] is a peer review web application created at the University of Melbourne. It is only available for users of the LMS at the university. Inside the web application, the instructors can create peer review assignments of different kinds. As with PeerMark, it is possible to let the software itself do the review allocation between the students, but it is also possible to specify rules such as:

- Reviews based on classes (modules inside a course)
- Reviews based on groups (teams of students)
- Reviews based on topics in a course

The instructors can also create some guiding questions to the reviewers, but it is not possible to create more formal criteria for the reviewers to give points based on as in for instance Canvas or PeerMark. After the reviews are given, the submitters can read the review of their own work, before they can have the opportunity to give a feedback on the review.

3.2.3 CPR

CPR [57] is also a web based tool for creating assignments to be peer reviewed by others. CPR stands for Calibrated Peer Review, and the reason behind this is that it focuses on preparing the students for the peer reviews. This is accomplished by making them review three calibrating assignments created by the instructor, to see if they fulfill the acceptable standard for conducting peer reviews on real student submissions. This is the major difference from the other peer review applications, that does not focus on preparing the students, but rather take for granted that this is something they already know how to do. As one can see in Figure 3.7, the graphical user interface is somewhat old, but the idea behind using calibrating peer review assessments to prepare the students still stands as very interesting.

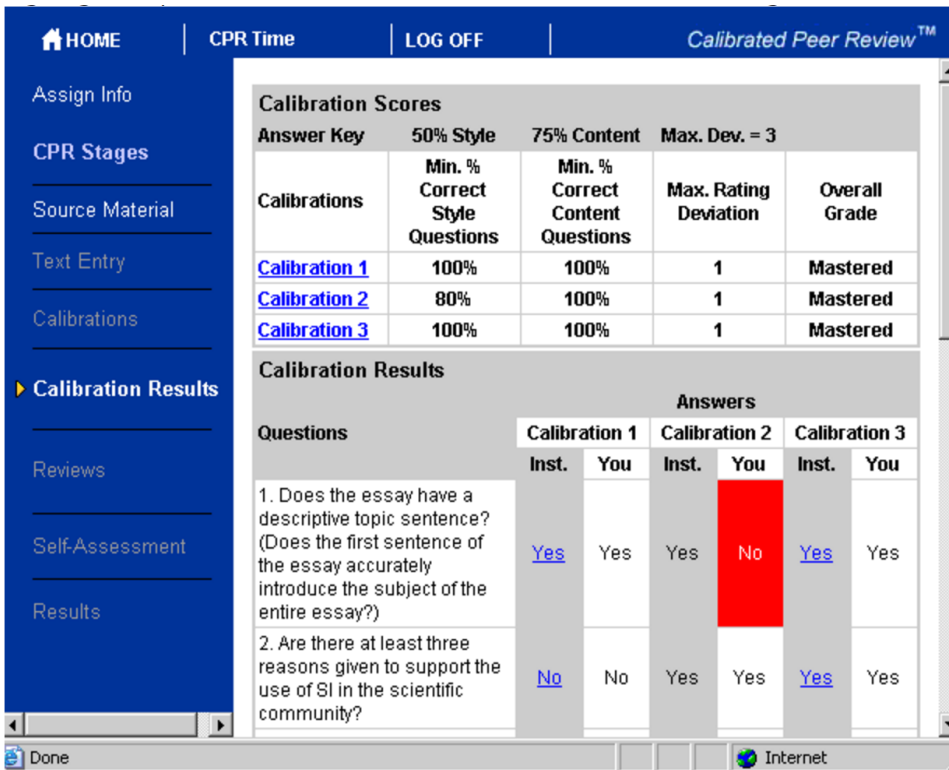


Figure 3.7: Calibrating peer review example in CPR [58]

Interviews

In order to gain first hand knowledge on peer evaluation, usage of LMS, and digitization, some interviews were conducted. The interviewees were mainly professors at NTNU who had an interest in the topics in question, and who wished to share their experience and knowledge on the subjects. This was a qualitative research, and the data gathering was important in order to try to answer the research questions as thoroughly as possible [Section 1.1].

4.1 The interview process

The interviews followed a fixed procedure where they took place face to face and with a given set of terms. The interviewees had to be informed about the usage of the qualitative data gathered in the interviews; that it was going to be handled in an ethical way and only used in the master thesis as well as that they were to be kept anonymous.

It was also important that they were presented the problem description of the project. More specifically, it was elaborated about the goal of creating a digitized version of a peer review assessment in Blackboard. In addition, it was presented the importance of their contribution to the outcome of the project, and that there is a high potential in creating digital modules to Blackboard which can be tailored to end-user needs.

4.1.1 Type of interview

Semi-structured interviews was chosen as the most appropriate interview type, as these types of interviews are setup to give the candidates the possibility to speak more free, and at the same time feel comfortable. Open interviews was also a possibility but was not used, as it could potentially give to much irrelevant information, thus choosing semi-structured interview as interview method was the key to get open-ended question giving

us the opportunity to identify hidden aspects of the candidates, hence giving us a deeper understanding of the topic.

4.1.2 Interview questions

```
# Q1: Have you had any experienced with the use of the peer review
  ↳assessment method, and if so what was your experience of it?

  # Q1.1: If only positive/negative - do you see any positives/negatives
    ↳ of these kinds of systems?

  # Q1.2: Would you trust the students to evaluate each other? Please
    ↳elaborate.

  # Q1.3: Would you use peer reviewing as an evaluation method that
    ↳affected their grades or just as normal exercises with approved/
    ↳not approved?

# Q2: If a digital (web page or blackboard plugin) peer evaluation module
  ↳were created, is there any specific task types that you feel could
  ↳make use of such a module?

  # Q2.1: Which aspects would be important to include from this sort of
    ↳task type?

  # Q2.2: Is there any general functionality that would be smart to
    ↳include in this module (such as giving evaluation directly
    ↳inside the browser, instead of downloading file then upload with
    ↳ comments)?

  # Q2.3 Would you utilize such a module in any of your courses?

# Q3: Please go through a scenario were you would use peer assessment or a
  ↳ similar evaluation method.

  # Q3.1: Were there any difficulties with such as scenario?

# Q4: If you were to request a digital module for yourself (as a professor
  ↳) is there anything you would like to see implemented for use in
  ↳your class?
```

Question themes

The initial question (Q1) is to gather knowledge on the professors experience of peer review. That is, what positive or negative effect it had, how the students experienced and handled it, in addition to any changes they would apply. Question Q2 is directly linked to research question 2 [Section 1.1] as it tried to extract what features a peer review module needs. It is also important to gather different points of view from the different professors as the features needed can change based on the course. In the following question Q3, a scenario was evaluated were professors used peer assessment. This included pinpointing the places in the assessment process that were difficult to achieve, thus making room for improvement at these points. The final question Q4 was created to identify and map the

needs for tailored software modules. This, to illustrate the importance of understanding how to develop custom modules to LMS.

4.2 Interview results

After conducting all the interviews, a lot of qualitative information was gathered. In this section, the answers from the interviewees are presented in a structured and summarized way. In the last part, a list of specific functionality for the development of a peer review module are given. This functionality is extracted from the answers of the interviewees, and does only represent a subset of the functional requirements of the peer review module that was developed.

Q1 - Earlier experience

The interviewees had a mixed level of earlier experience with the peer review assessment method, from little or zero experience, to the use of an actual system incorporating student peer evaluation. The ones with experience with this kind of method, were positive to it. It is of course positive for the enhanced learning of the students, but also that it could potentially save time for the professors as well as lowering the need to hire many teaching assistants.

There are also some aspects one has to consider to ensure good quality on the feedback given to the students. One possibility is to make more than just one student give a feedback on a delivery and use the average score given by the different students as a guiding score for the teaching assistants or professor. When questioned on whether or not they entrusted the task of grading to the students the interviewees indicated that they did trust the students to give an evaluation score, but that it is not legal for them to give scores that affected the grades of the other students without the professor or teaching assistant accepting the score given as a final step. It would be important to make this step as little cumbersome as possible, especially in cases where there are many students to be reviewed.

One interviewee mentioned that the use of peer review could be used on approved/not approved exercises, and that the students could then file a complain if they were not satisfied with the outcome, which in turn the teaching assistants can review to give a final answer. It is important to note that one would need some kind of mechanism for checking the exercises that are already approved, simply because these can also be wrongly reviewed. An indication on a wrongly approved exercise can for instance be a lack of text, and this is something that the teaching assistants should be notified about. After the teaching assistants are notified they should give their own review of the submission.

The interviewees mentioned that it, in general, is harder to motivate the students to do peer reviews than normal assessment, as they does not see that much value in it. A potential solution to this would be to highlight the fact that these types of reviews are common to conduct when one enters employment. This has to be highlighted during lectures, and one

can even hire guest lecturers to give practical examples.

Another aspect that can be demotivating for the students is if the assignment itself is disorganized. Thus, the use of such a method has to be well planned in addition to simple and clearly. It will be important for a peer review module to only show the most important parts to the students in order for it to be convenient to use.

Q2 - Task types

One of the interviewees mentioned that it would be nice to have support for code review in programming courses. So it should be possible to give exercises displaying an exercise description including both normal text as well as code. It should also be possible for the students to deliver their code and for the evaluators to see the code in an orderly way, for it to be as little cumbersome as possible to review it.

Another suggestion was to create a back-end servlet that could be used to connect to the current code evaluator of the courses that uses this, and then focus on presenting the results of the code in good manner to the students.

One could also make use of peer review in task types that normally are voluntary with zero to little feedback. This way, the students will have a reason to do the exercises more than just doing them without getting a proper feedback on their own deliveries. It would be important to find a good way of assigning the evaluator pairs, so that for instance a A student does not get a D or E student as an evaluator.

It would be important to go through examples and give thorough tutorials on how to do good peer reviews. This is something that should be done during the introductions lectures, to prepare the students. One of the interviewees also mentioned that the anonymity of the students is something that would be important, as there had been cases where students that had evaluated other students had been prompted to explain themselves, because the submitter had not been satisfied with the result.

An additional aspect that would be important to include is the use of guiding criterion, for the evaluators to follow when they are reviewing the exercises. This to heighten the quality of the feedback as well as making sure that the evaluators knows what they should look for, and hence the feedback's will be more uniform. The criterion needs to be unambiguous and easily accessible for the users, which should include the possibility of illustrating both good and bad examples of the usage of the criterion, as well as giving links to for instance slides, explaining the criterion in more detail.

Another interviewee had the need to make it possible to create peer review assessments that could connect students taking different modules inside one course. For instance if one student had chosen a module or topic regarding object-oriented programming and another student a topic regarding functional programming, it should be possible to create the peer assessment inside both these modules and still be able to connect these students

assessments. The interviewee also had other concrete suggestions to functionality that should be included in a peer review module:

- Rubric criteria.
- Assignment and peer review creation in one module.
- Evaluation screen must contain both the submission and criterion's.
- Easy and intuitive for the evaluators to give points.
- Visual representation of the submissions.

Q3 - Scenario

Some of the interviewees had examples of scenarios where they would make use of a peer review software.

BPMN modelling

One of the interviewees had made use of such as system in a course were the students were to create process models, through the use of Business Process Modeling Notation (BPMN). The students did get the same overall exercise, to model a process, but which process was randomized between the students. Each student delivered their model into the system, before each model was assigned to three different other independent students for evaluation. This to heighten the probability of at least one good and thorough feedback. The chosen students did not have the same process to model in their exercise, thus they had to familiarize themselves with the new process model as well as their own. The reasoning behind this was that the students might get blinded by their own process model if it was the same as the evaluation model.

After the evaluation was given, the students were to refine their model based on the feedback from the others. There were of course some cons of this use of the peer review method - because the students had to evaluate a different process than their own as well as the fact that they had to do three times as much evaluation work then the original exercise resulted in a lot of work for one exercise.

Programming

Another scenario that we got from another of the interviewees was exercises containing programming and code review. The students would deliver an exercise where they would solve some programming tasks. When they delivered, there would be an evaluator-algorithm of some sort to automatically evaluate the code. But, in cases where the code wasn't possible to run for the algorithm it would automatically give 0 points to the student. In those cases, it would be an idea to make use of peer reviews so that the students at least got credit for some of their work. The students that were to review the code would be those who had already delivered the exercise and gotten a good score on it (for instance more than 90% correct), and if the student did not agree with the outcome, he/she could

complain and a teaching assistant would do a final review. The student reviewers would have to get some extra points, as an extrinsic motivational factor.

Q4 - Other modules

In this question, the interviewees had a range of different ideas.

Best questions

A module where the students can ask questions about course topics. Other students, teaching assistants and professors can then answer these questions as well as giving them up-votes. This can also be done to the answers to these questions. As a whole, this results in a some sort of forum solution, like stackoverflow [59]. The only difference is that it will be integrated into Blackboard. This would also make it easier for later students to identify and understand the most important aspects of the course based on earlier questions and answers. One potential negative aspect of such a module would be that the people who answers the questions may disagree on what is the most correct answer.

Learning goals

A module that supports a more detailed listing of the different learning goals in a course. Right now, the learning goals are often very abstract and vague, and by introducing such a module, it may be easier for the instructors to define more explicit goals. These goals could then be linked up to the different exercises as well as for instance the custom flash-cards mentioned earlier.

Custom Flashcard

Here, the main idea is that the professors or teaching assistants creates a deck of cards containing questions and answers from different topics of the course. The students can then practice by going through these decks of questions. These kinds of solutions already exists in many forms, from third party mobile and web applications to standard quizzes in learning management systems.

For the students, they should have one deck that contains all the decks from the different courses they are taking. They could then be prompted with something like: "Today you have 15 cards you should look at". These cards should then correspond to where the different courses are in the curriculum. One should also make sure to include functionality such as spaced repetition, where the students does not get the same questions over and over again if they already have answered them correctly. An example of such a flashcard application can be seen below in Figure 4.1

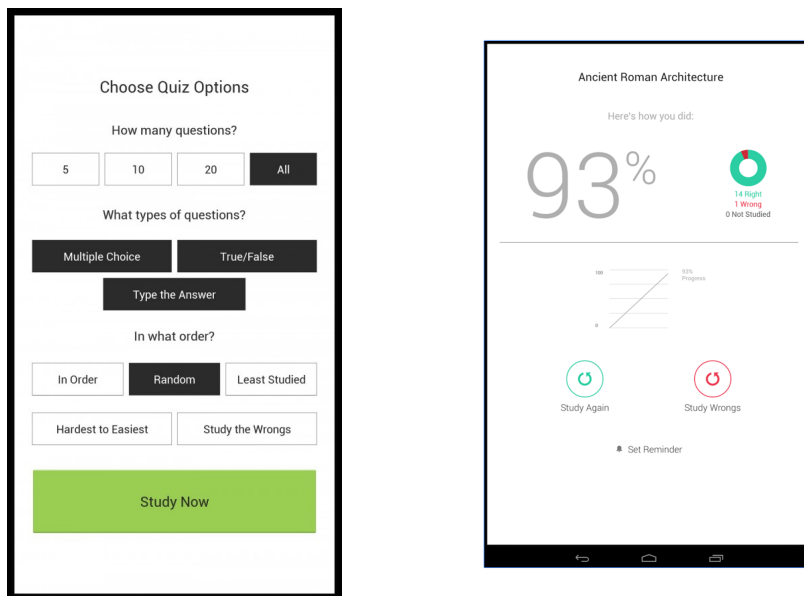


Figure 4.1: Flashcard example - StudyBlue Android application [60]

4.2.1 Summary - Identified requirements

From the interviews, a lot of different information was gathered. The essential parts, focusing on explicit and implicit functionality is stated in this subsection. This functionality is reflected in Chapter 5 - Design as well, specified in a more detailed manner in the requirement specification. Even though these interviews were used as an essential part of the design foundation in this project, it is important to note that they only give an indication on how to make use of and create a peer review module in a university context, and that this is only one out of many possible solutions. To get a clearer answer, more interviews have to be held, both with instructors, but also with other end-users such as the students.

- **Average score**

The teaching assistants and instructors should be displayed the average score from all the reviews of a submission.

- **Accepting score**

It should be an easy way for the instructor and teaching assistants to accept or adjust the average score.

- **Convenient module**

The module itself has to be orderly and convenient for the students to use. Thus a high usability and a simplistic presentation is important to motivate the users.

- **Pairing of evaluators**

There should be an algorithm for pairing the evaluators in a good manner - meaning that all the evaluators should for instance have an equal amount of reviews to do.

- **Anonymity**
It would be important to have the opportunity to make the submitters and evaluators anonymous. This to avoid potential conflict.
- **Criteria**
To make the reviewing process as qualitative as possible, it would be essential for the instructors to have the possibility to create guiding criteria.
- **Connect cross-module students**
Students in different modules within a course should be able to evaluate each other.
- **Number of evaluators**
When creating a peer assessment it should be possible to specify the number of evaluators.

Chapter 5

Design

Good software design is about conceptualizing the user requirements and making a plan to find the best possible design for implementation of the intended solution. In this chapter, the results from the theoretical research is put together with the feedback from the interviews and knowledge from state of the art, into a software system design.

The importance of software design comes from the fact that preparation before implementation can mean the difference between a good and a bad software. Hence, before starting the implementation process a target group was determined, a user interface was defined, the software requirements were identified and architectural views was created based on the stakeholders different views of the software.

5.1 Design Goals

In this paper there were two goals connected to digitizing peer reviewing. The first was to increase the understanding of peer reviewing as a learning tool and how it is used in today's education, while the second was to extract features needed to build a digital peer review module. These goals were accomplished theoretically by reading papers describing the use and purpose of peer reviewing in education, and by viewing how state of the art peer review software was utilized. The practical part was covered by interviewing instructors, detailing the current usage of peer reviewing at NTNU and defined how it is being used in current education.

The knowledge gained from the two goals was required to design and develop the digital peer review module (e.g. identifying which features that was required or wanted by instructors). This development of the module covered both RQ2 and RQ3 while also taking into account the knowledge of RQ1. As an example; when creating the peer review module one needed to determine important features of such a software (RQ2). Secondly, one also needed knowledge on how it could be utilized in an university context, in order for it to appropriately function in current education (RQ1). And lastly, when developing such

a software one could potential encounter bottlenecks with the Blackboard development platform that had to be documented for future reference (RQ3).

In Figure 5.1 below, an overview of the peer review module is given. The figure illustrates how the instructor and students are to use the module as an assignment is created by the instructor, answered and submitted by a student, before it is evaluated by other peer students. This illustration only displays the core functionality, namely the standard peer evaluation process. More functionality like evaluation of evaluations and group evaluations are also available.

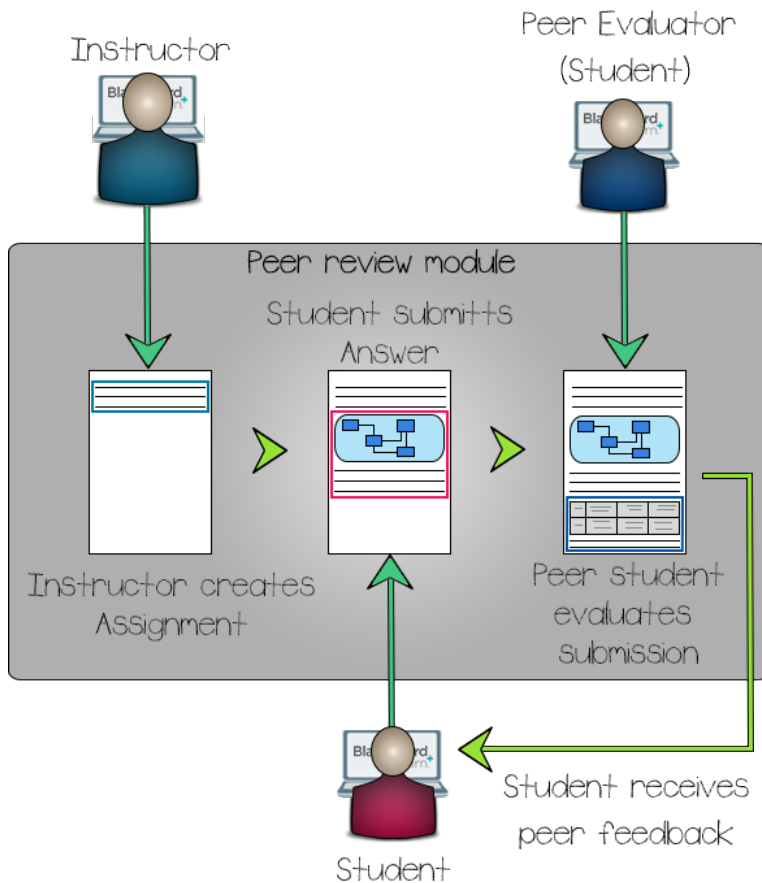


Figure 5.1: Illustration of the peer review process using the peer review module

5.2 Stakeholders

Stakeholders are people that in some way, have an interest in the software developed. The stakeholders are often split into general groups of people that describes their connection to the product. When developing a software, it is always important to have the different stakeholders in mind, as they are the ones influencing it as well as being affected by it.

5.2.1 Stakeholder concerns

In this subsection, the general stakeholders are presented together with their main concerns when it comes to software architecture and quality attributes. This, to understand the different aspects that needed to be in focus during the development. The most important quality attributes are usability, modifiability, and security requirements as these focus on the end users and developers. The interoperability, availability and testability attributes are less prioritized, as they are partly not within our control, or because it is outside the scope of the project (e.g. the module is highly experimental, so its focus is not on being 100% testable). Later in Section 5.4.2 the usability, modifiability, and security requirements are explained in detail.

Stakeholder	Concern
Developers	<i>Modifiability</i> : It should be easy to make changes to the implementation.
Server and database owners	<i>Interoperability</i> : The product should communicate information in a meaningful and good way.
End users	<i>Usability</i> : The product should be easy to use and quick to learn. <i>Availability</i> : The product should always be up and running for the users to have a good experience. <i>Security</i> : End users should not be able to access personal information of other users.
Testers	<i>Testability</i> : It should be easy to test the different parts and features of the system

Table 5.1: Stakeholders

5.2.2 Target group

The target group is the group of stakeholders that the software is designed towards. A software developer always has to keep all its stakeholders in mind when designing a software, but one or more stakeholders may be prioritized over others, and for this software, the end users had the highest priority. This is the group of stakeholders that the software is intended to be directly used by. As the software is designed with these in focus, their opinions and feedback are important in virtually all the phases of the software development

process. The end users for the peer assessment software is the instructors and students that uses the Blackboard LMS, but also actors such as teaching assistants and collectors of the evaluation data.

As this was part of an experimental development paradigm, we chose to have a main focus on the instructors, as they had potentially most of the knowledge regarding core functionality needed in a peer review module. Nevertheless, it is necessary to highlight that having the students in mind during the development was still important. This, to ensure that the software was motivational and easy to use for them as well.

Instructor

The instructors are the ones responsible for creating the peer assessment, educate the students in the use of peer review and ensuring that the whole process gives the student the best possible learning outcome. They are concerned with the creation of the peer assessment and details of the assignment. It is also important that the process of distributing and evaluating the assignment is as streamlined as possible.

Student

The students are responsible for doing, delivering and reviewing assignments. They are also the ones that are supposed to have high learning gains. A system must take into consideration that the students may be less experienced in both peer reviewing and in using Blackboard modules. The way the system presents the assignment and the intuitiveness of the user interface can be the key factor of whether the students enjoy the assignment or not.

5.3 User interface

The user interface is the main interacting layer between the software and its users. A good user interface is intuitive, easy to use, attractive and has a consistent design. One could also say that it needs to be maintainable (e.g capable of handling changes in features), and have a quick response time. When designing a user interface one needs to be agile, because the user interface evolves with the software and the changes done to it. Even so, it is still important to have a vision from the beginning of the development process and take into considerations things like the limitations of the software, knowledge of how similar software UI looks, and general guidelines.

5.3.1 Guidelines

When designing a user interface it is important to follow some basic principles and guidelines of design. The reason for this is to learn from previous success, and to avoid doing mistakes that could have been avoided with more knowledge on design guidelines. For this project it included looking at previously made interfaces from state of the art software presented in Chapter 3 - State of the art, using previously identified design rules like the

eight rules of Shneiderman [61], and looking at articles on good web UI design [62, 63].

From the state of the art, there were some functionality and design choices that inspired the design and development of the peer review module in this project. For instance, the evaluation view of the module is very much inspired by the one used in the Canvas LMS and PeerMark, which makes use of comments as well as evaluation criteria and Crocodoc for display of uploaded files. The Crocodoc display is also already in use in the assignment module in Blackboard. Even so, it was not possible to make use of Crocodoc in this project, as it was not available anymore, and hence ViewerJS, a similar software extension was used as a replacement.

Shneiderman's eight golden rules of interface design are eight simplistic rules that describes design aspects that needs to be considered when creating a software module to increase the usability for the end users. For instance, by having a consistent design, with all the buttons looking similar for similar functionality, one avoids confusing the user. A more detailed description of quality requirements striving to fulfill these golden rules is presented in Section 5.4.2.

5.3.2 Blackboard theme

Blackboard as an LMS has a very distinct theme. And hence, Blackboard is offering a tag library that can be used instead of standard HTML that has a predefined styling in order to make the modules created for Blackboard follow the same theme (Figure 5.2). The theme itself is a little bit cumbersome, and some of the styling and UI elements does not follow best practices for UI design. For this project's peer assessment module the Blackboard tags was used, but some custom added styling was applied to increase the usability and to better follow design guidelines.

Blackboard bbNG tags	HTML Tags
<pre> <bbNG:learningSystemPage> <bbNG:pageHeader> <bbNG:pageTitleBar title="{page_title}"/> </bbNG:pageHeader> <bbNG:form > <bbNG:dataCollection> <bbNG:step title="ASSESSMENT INFORMATION" id="assessment_name"> <bbNG:dataElement label="Assessment Name"/> </bbNG:step> </bbNG:dataCollection> </bbNG:form> </bbNG:learningSystemPage> </pre>	<pre> <head> <title>\${page_title}</title> </head> <body> <form> Assessment Name:
 <input type="text" name="Assessment Name">
 <input type="submit" value="Submit"> </form> </body> </pre>

Figure 5.2: Blackboard bbNG tags versus HTML tags

The general design of Blackboard is built around folders and forms, and is in many ways very static compared to the dynamic web design one often use today. NTNU is as stated

earlier, implementing Blackboard as their main LMS and in conjunction with this, they hired UI/UX designers to look at the interaction design aspects of the Blackboard platform. The results from these designers pointed out that there were many aspects of the Blackboard web design that was outdated and easy to misinterpret. Some design flaws that were mentioned were:

- Uncertain what is clickable (bad affordance) (Figure 5.3)
- Unclear importance of elements
- Design is perceived as messy
- Each action requires many clicks

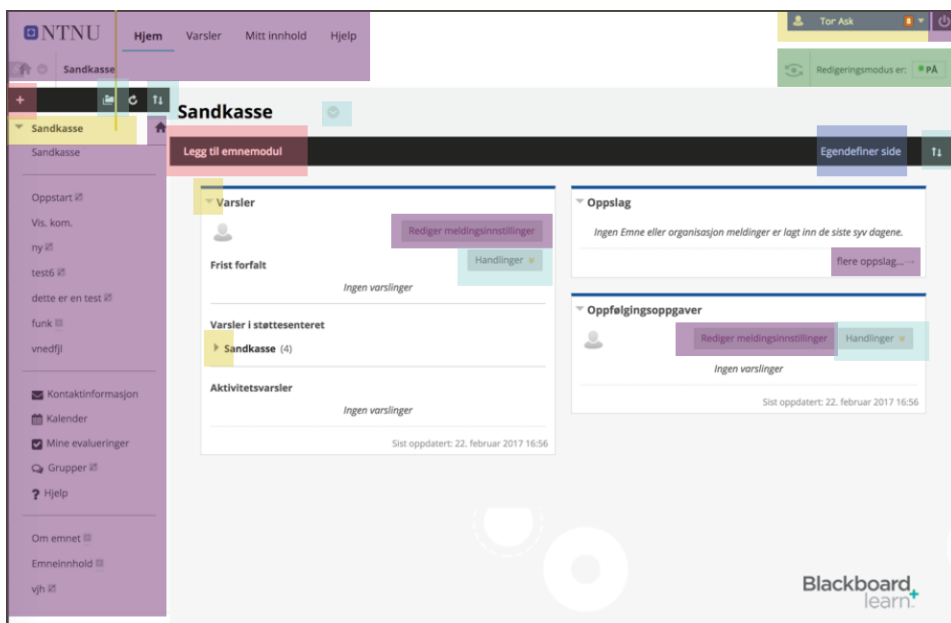


Figure 5.3: A standard Blackboard page, with highlights on different clickable areas¹

Overall, there were many aspects of the Blackboard theme that had a potential for improvement. Thus, this was something which was important to take into account when designing the peer review module and is discussed further in the next section.

5.3.3 Chosen interface

Based on the information gathered in the interviews, the guidelines and the Blackboard design, some sketches were made for this project's design. The first part of the design revolves around the creation of an assessment, where a comparison between the suggested

¹Taken from design project by BEKK

design against the current peer assessment module in Blackboard is given. The second part explains the design choices for evaluating an assessment, which is the most important aspect of the module. In the final part, the general improvements to the design are discussed.

Assessment creation

In the current Blackboard peer assessment module one has to go through a lot of views and clicks. The module also has many advanced options like creating multiple questions with separate criteria. This advanced approach gives many options, but according to our interviews none of these options had a high priority. Many instructors also questioned the usability of the module as it was difficult for first timers to create a simple peer assessment test. Figure 5.4 displays the current Blackboard peer assessment module and all the views involved in only creating a new peer assessment.

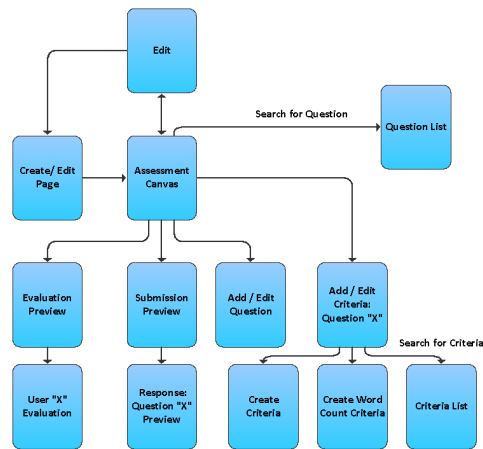


Figure 5.4: Assessment creation process page overview, from the current Blackboard peer assessment module

The design chosen for this project aimed to remove many of the advanced features and focused on a more streamlined and usable design. This included removing the ability to make separate questions as multiple questions could be stated inside the same instruction text instead. Another goal was to remove the amount of views and clicks in general and focus on the browsing strategy [Section 2.2.3], hence many of the separated views were merged into one single view. This gave a major reduction in both complexity and number of clicks required in order to create an assessment.

Assessment evaluation

While the creation of an assignment is mostly important for the instructor, the peer evaluation process is the most important aspect for the students. The peer evaluation process consists of selecting a student submission and evaluating it according to the given criteria.

Meaning that the student will analyze another student's submission, then write a feedback and give a score based on the evaluation criteria. The key in our design was to highlight the three main elements of an evaluation:

- Reading the student submission
- Understand the evaluation criteria and providing a score
- Writing and reading the feedback

One of the issues with a submission is that it can be both file and text based, and often one has to download the text before reading it. This increases the threshold and time it takes to analyze a submission, and can be dreary for the evaluator. In order to streamline this process, the submission is displayed using ViewerJS [64]. ViewerJS allows the student to see the submitted text inside the browser, relieving the student from having to download the text to the computer.

In order for the student to understand the evaluation criteria, it needs to be clearly explained, thus a location where the criteria are written need to be provided [Section 2.1.2]. According to design guidelines good design should also reduce the short term memory load (e.g. you should not have to remember what was on the previous page)[61]. As a result of this, our design displays the criteria on the same page as the student submission and the feedback input. The criteria or guidelines for writing a well defined feedback should be easy to implement and use, as stated in Section 2.1.3, so the design uses rubrics as an option for written criteria. By using the rubrics, the students can simply select a valid score for each row based on how well the submission meets the criteria. This makes the whole process of both scoring and evaluating more unified for all students. According to Section 2.3.3 about goals, the use of guiding criteria is important for the intrinsic motivation, thus this was an important functionality to include.

Lastly, it should be easy to write a feedback on a submission. To solve this, the ViewerJS view of the submission and a rubric with criteria are displayed on the same page as where the feedback is to be written. And only by clicking one simple button, the feedback can be sent. But it is not only the beginning of evaluation process that needs to be of priority. The way the feedback and criteria are displayed, also matters for the reader of these evaluations, including both the instructors, who can create an additional evaluation of their own, and the submitter of the original assignment. In order to make this as streamlined as possible, the evaluations are displayed as separate "messages" descending down the page. This makes it easy for the student to see the evaluations from both the peer students and instructors, and makes it flexible to add as many evaluations as needed without adding complexity for the students.

Figure 5.5 displays an early sketch where ViewerJS is set to display the submitted PDF, and instructions from the teacher is displayed above the evaluation part. In the evaluation part, the evaluator has a rubric with criteria set by the instructor, and a text field where they can write a detailed feedback on the submission.

<u>Student name</u>	<u>Assignment name</u>	<u>Evaluation due date</u>
<div style="border: 1px solid black; padding: 20px; margin: 10px auto; width: 80%;"> <p style="text-align: center;">Student Submission (PDF, Etc)</p> <p style="text-align: right;">ViewerJS</p> </div>		
<hr/> <p>Criteria Instructions</p> <hr/> <p>- Instructions</p>		
<p>Criteria Rubric</p> <hr/> <p style="text-align: center;">Evaluation</p>		
Criteria	Scales	
Criteria 1		
Criteria 2		
<hr/> <p>Feedback</p> <hr/> <div style="border: 1px solid black; padding: 20px; margin: 10px auto; width: 80%;"> <p style="text-align: center;">Text Field with feedback on the submission</p> </div>		

Figure 5.5: Early sketch of evaluation view(as seen by the evaluator)

General improvements

In addition to the larger improvements to the assessment creation and evaluation process, there were some minor improvements made based on the interview feedback [Section 4.2.1]. From the interviews it was clear that average score and score acceptance should be easily available, in order to make the scoring process as streamlined as possible for teaching assistants and instructors. To solve this issue the list of the student submissions included an average score based on the peer evaluations given. The list also contained submit buttons for correcting a single submissions directly, thus enabling the instructor to approve all submissions without having to enter them one by one. This feature is crucial for an instructor with hundreds of students, where reviewing the submissions one by one could take a massive amount of time. Figure 5.6 illustrates how the submission list would look like.

List displaying all the submissions with the average score to each

<input type="checkbox"/>	Name	Submission reviews	Avg. Score	Fin. Score	Status
<input type="checkbox"/>	Student 1	Review Submission	10/22	<input type="text"/>	<input type="button" value="Submit"/> Not corrected
<input type="checkbox"/>	Student 2	Review Submission	14/22	<input type="text" value="14"/>	<input type="button" value="Submit"/> Corrected

Figure 5.6: Illustration of a list over student submissions with evaluation scores ready to be given the final score

The interviews also revealed other issues where for example the submitter of an assignment confronted one of the evaluators, and one solution to this would be to make the process anonymous. Another request was to be able to specify the number of evaluators that should evaluate a submission. These things might be minor check-marks in the user interface, but plays an important role for the peer evaluation process as a whole (Figure 5.7). Our user interface aimed to include as many changes as possible that would help simplify and streamline the whole peer assessment process and the design features that were chosen did just that.

Number of Submission to Evaluate:

Allow Anonymous Evaluation: Yes No

Figure 5.7: Illustration of evaluation numbers option and anonymity option that would be added into the assessment creation form

5.4 Requirement specification

This section discusses the various requirements for the system, more specifically functional requirements and quality requirements. Functional requirements focus on which tasks the system should be able to perform, such as being able to create an assessment. The system needs to fulfill certain qualities, such as performance or usability, called quality requirements.

5.4.1 Functional requirements

The functional requirements focuses on the specific tasks the software must handle. They list the requirements that the software must or should have. All the requirements are graded according to priority, with the scale going from mandatory, significant improvement to optional. The requirements that are marked with mandatory are functionality that is required in order to make the software viable to its users. Significant improvement and optional requirements are implemented based on time left after mandatory requirements and implementation difficulty. The following tables describe the functional requirements for our software.

- **FR Name:** Name and number of the functional requirement.
- **General description:** Details what the requirement entails.
- **Priority:** Grades the requirement based on importance.
- **Objective:** What the fulfillment of the requirement accomplished.
- **Context:** Why this requirement is required.

List 5.1: Table fields

FR1	Peer assessment creation
General description	The user should be able to add/edit the peer assessment name, instructions, files and submission dates in the module
Priority	Mandatory
Objective	Let the user create a new peer assessment and add a name, instructions, files and submission dates to it. Also, later on the user can edit this.
Context	-

Table 5.2: Functional Requirement 1: Peer assessment creation

FR2	Point specification
General description	The user should be able to specify the maximum amount of points possible to get from the assessment
Priority	Mandatory
Objective	Let the user create a new peer assessment and add maximum amount of points possible. Also, later on the user can edit this.
Context	-

Table 5.3: Functional Requirement 2: Point specification

FR3	Evaluation dates specification
General description	The user should be able to add/edit the start and end date regarding the evaluation of the submissions in the module
Priority	Mandatory
Objective	Let the user create a new peer assessment and add start and end date for evaluation of the submissions. Also, later on the user can edit these dates, as long as it is before the start date. The evaluation start date has to be after the submission end date.
Context	-

Table 5.4: Functional Requirement 3: Evaluation dates specification

FR4	Assessment criteria creation
General description	The user should be able to create and add a rubric as guiding criteria for the evaluation in the module
Priority	Mandatory
Objective	Let the user create a new peer assessment and add a rubric with criteria for evaluating the submissions. Also, later on the user can edit these criteria, as long as it is before the start date of the evaluation
Context	Taken from identified requirements in the interview phase

Table 5.5: Functional Requirement 4: Assessment criteria creation

FR5	Number of evaluations
General description	The user should be able to set the number of evaluations to conduct for the students in the module
Priority	Mandatory
Objective	Let the user create a new peer assessment and specify the amount of evaluations to conduct. Also, later on the user can edit this choice, as long as it is before the start date of the evaluation phase
Context	Taken from identified requirements in the interview phase

Table 5.6: Functional Requirement 5: Number of evaluations

FR6	Student evaluation display
General description	The students should be displayed a list of the submissions they have to review
Priority	Mandatory
Objective	When the evaluation interval begins, the students should be able to see the submissions they have to evaluate in a list
Context	-

Table 5.7: Functional Requirement 6: Student evaluation display

FR7	Grader submission display
General description	The graders (instructor or teaching assistants) should be able to see a list of all the submissions with detailed information connected to the reviews of these submissions
Priority	Mandatory
Objective	When the submission period is over, the graders should be able to see a list of the submissions delivered, as well as information connected to peer reviews conducted on these submissions
Context	-

Table 5.8: Functional Requirement 7: Grader submission display

FR8	Peer review comment
General description	Peers should be able to write a comment when reviewing a submission
Priority	Mandatory
Objective	When the evaluation period has started and the peers are to review each other, they should be able to add comments to their reviews
Context	-

Table 5.9: Functional Requirement 8: Peer review comment

FR9	Set assignment type
General description	The user should be able to specify if the assignment is individual or by groups
Priority	Significant improvement
Objective	Let the user create a new peer assessment and define if the assignment is individual or by groups for the delivery. Also, later on the user can edit this.
Context	-

Table 5.10: Functional Requirement 9: Set assignment type

FR10	Anonymous student evaluation
General description	The user should be able to make the evaluation process anonymous in the module
Priority	Significant improvement
Objective	Let the user create a new peer assessment and specify evaluation anonymity. Also, later on the user can edit this choice, as long as it is before the start date of the evaluation phase
Context	Taken from identified requirements in the interview and PeerMark in state of the art

Table 5.11: Functional Requirement 10: Anonymous student evaluation

F11	Student self evaluation
General description	The user should be able to specify that the students should do a self evaluation in the module
Priority	Significant improvement
Objective	Let the user create a new peer assessment and specify self evaluation for the students. Also, later on the user can edit this choice, as long as it is before the start date of the evaluation phase
Context	Blackboard in state of the art

Table 5.12: Functional Requirement 11: Student self evaluation

FR12	Automatic peer evaluation pairing
General description	The system should be able to automatically assign the peer evaluations
Priority	Significant improvement
Objective	Let the user specify if they want automatic evaluation pairing - the system should then pair up the peers
Context	Several modules in state of the art

Table 5.13: Functional Requirement 12: Automatic peer evaluation pairing

FR13	Interactive rubric for evaluation score feedback
General description	Peers should be able to use an interactive rubric to give score base on criteria
Priority	Significant improvement
Objective	When the evaluation period has started and the peers are to review each other, they should be able to give score to the submissions through the use of an interactive rubric
Context	State of the art - Canvas and Blackboard inspired

Table 5.14: Functional Requirement 13: Interactive rubric for evaluation score feedback

FR14	Peer review of peer review
General description	Submitters should be able to give feedback and review the reviews on their submission
Priority	Significant improvement
Objective	When the evaluation period is over, the submitters should be able to give feedback on the reviews of their assignment submission
Context	PRAZE in state of the art

Table 5.15: Functional Requirement 14: Peer review of peer review

FR15	Calculate average score of peer reviews
General description	Average score of the reviews should be displayed in the graders submission display on each submission in the list
Priority	Significant improvement
Objective	In the graders submission view, each submission should be displayed with an average score of the reviews
Context	Taken from identified requirements in the interview phase

Table 5.16: Functional Requirement 15: Calculate average score of peer reviews

FR16	Edit and accept evaluation score
General description	Graders should be able to accept or change the score displayed in the graders submission display
Priority	Significant improvement
Objective	In the graders submission view, each submission should be displayed with the possibility to accept or change the average review score connected to it. This should then be sent to the evaluation center in blackboard
Context	Taken from identified requirements in the interview phase

Table 5.17: Functional Requirement 16: Edit and accept evaluation score

FR17	Delegate grading
General description	The user should be able to delegate the grading to others then himself when creating an assessment
Priority	Optional
Objective	Let the user create a new peer assessment and specify other users that should be able to grade the submissions and evaluations. Also, later on the user can edit this.
Context	Blackboard in state of the art

Table 5.18: Functional Requirement 17: Delegate grading

5.4.2 Quality Requirements

There are many factors that needs to be in place in a system that is to be used by a large number of users. Things like usability where the ease of using the system interface plays a big role, is essential when introducing the software to new users. There are also many risks involved with a system full of user information and requirements around things like security is therefore also important. Quality requirements addresses these issues which may not have a specific software feature in mind but rather a more general description on how the system should behave.

Usability

Usability is essential for satisfying the end user, and for making it easy for new and existing users to enjoy the software. This is accomplished by making their experience using the software as satisfactory as possible. It will also be important to help the user avoid mistakes such as accidentally clicking the wrong buttons. It was focused on following most of Shneiderman's Eight Golden Rules of Interface Design, to create a good user experience [61].

- **Conventions**

With conventions as confirm/back buttons it is easier for the end user to use the different aspects of the module, thus increasing the usability.

- **Consistent design**

By using the same colors on all the buttons, the usability is increased, since consistency leads the user to believe that all similar buttons function the same way. There was also a focus on highlighting clickable links and making the path the user should take more obvious.

- **Simplistic views**

As mentioned earlier, a lot of advanced functionality and cumbersome views with a lot of information display was cut. Instead it was introduced a lesser amount of views with a more simplistic display.

Security

Security is very important for not violating the privacy policy and for not making it possible for anyone to obtain private and sensitive information about specific users. In a Learning Management System there is often a large number of users and also a lot of sensitive information that needs to be shielded for public view.

- **Information exchange**

By only using internal calls and the Blackboard API for information exchange, it is ensured that the information is handled in a correct way.

- **Private information**

User sensitive information is stored in the already existing database solution used at NTNU.

- **Verified libraries and frameworks**

By using verified and known libraries and frameworks the software is more resistant to intruders and is also less prone to errors, thus giving the module higher security.

Modifiability

Modifiability is important for the developer and system maintainer to be able to do changes and add more functionality. In a system that is to be used by instructors and students alike, the ability to add new system features and modify existing functionality is crucial if the system wish to survive in the long run.

- **Splitting of modules**

By not creating big modules that contains a high diversity of different functionality, and rather splitting these modules into smaller ones, the system acquired a good modifiability. This is because changes in a big module, might affect other parts of it, as opposed to changes in smaller modules, where the ripple effects usually are much smaller.

- **Well-documented code**

By documenting the code thoroughly, it will be easier to perform changes later on. By commenting the different methods and classes, it will be less cumbersome to fix errors that can occur, especially for developers that has not written the parts that needs change.

- **Good object-oriented design**

A good object-oriented design increases the modifiability for the developer. This, as the design is uniform, thus it will be easier to do implementation changes later on.

- **MVC architecture**

Separation of presentation, abstraction and control has a similar effect as the splitting of modules point as mentioned earlier.

5.5 4+1 model

The 4+1 model is a view model designed by Philippe Kruchten as a way of describing the architecture of a software system. As the name indicates, there are four main views, in addition to one extra:

- **Development**

Static grouping of the development components in the development environment.

- **Logical**

Design overview of the object oriented modelling of the module.

- **Process**

Describes the system usage and synchronization of functionality.

- **Physical**

Reflects the software onto the hardware components.

- **Scenarios(extra)**

Illustrates the four other views through UML use cases.

The views are suppose to represent the different stakeholders, such as end-users, developers and project managers [65].

For our purpose the 4+1 model gives us a way of displaying multiple views of our system so that the readers of this paper can see things their way. It also allows us to illustrate the system in multiple coherent ways, which in turn makes it easier to understand how the system as a whole works.

5.5.1 Development view

The development view illustrates a system from a programmer's perspective and is concerned with software management. The view uses a layer diagram to describe system components (Figure 5.8). The diagram shows the peer assessment module's structure with Spring as the main underlying component used to connect the different parts together. By using Spring the JSP files represented by HTML and Blackboard Library Tags are able to send request with a given URI that is mapped to a corresponding controller from the Java modules. In the same way, the controller can send a response Object (e.g a ModelAndView Object) when triggered by a request from the JSP. The controllers contains methods for saving forms and creating ModelAndView objects, and is the heart of operations in the module. Beans are essentially helper classes for the controller and represents all classes which the controller uses to perform specific logic operations. In order to access, load and create database entities a Data access object (DAO) class is necessary. All DAO classes inherits a Blackboard parent DAO class which in turn runs the persist and load of table entities. Table entities that are created by the peer assessment module are represented in the code as its own table entity class. These are used by the corresponding DAO class for creating, updating and altering the entity in the database (e.g You got an Assessment class (Table Entity) representing the assessment table from the database, and when the controller wish to update the assessment it alters the variables of the Assessment class and uses an AssessmentDAO to update the assessment table in the database).

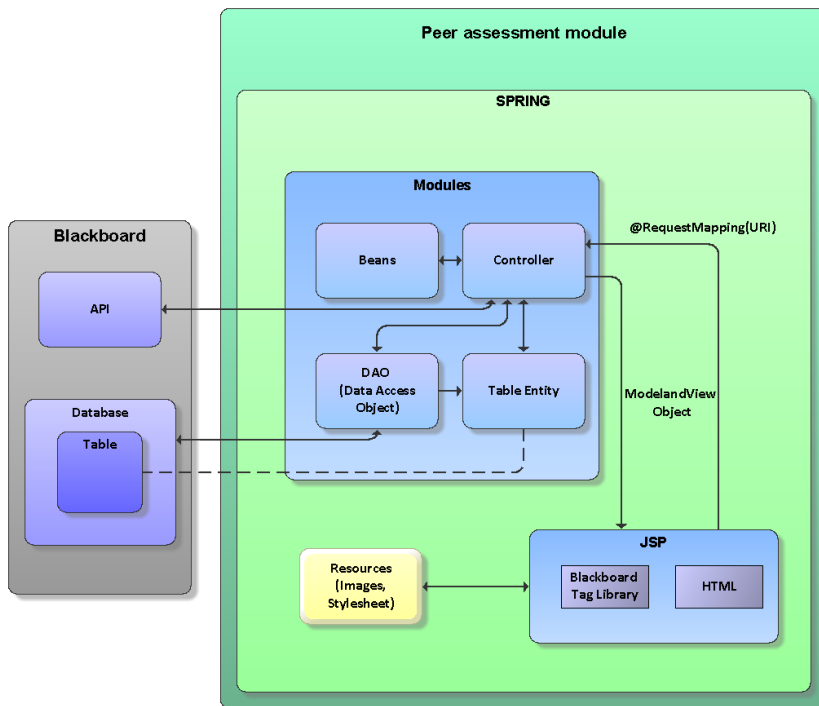


Figure 5.8: Layer diagram of the peer assessment module

5.5.2 Logical view

The logical view is concerned with the functionality that the system provides to end-users. It is represented as a class diagram containing the core functionality of the peer assessment module (Figure 5.9). In the diagram there are four main features; Assessments, Evaluations, Submissions and Reviews. There is also a ToolController for extra instructor functionality (e.g The instructor gets an overview over all Assessments options through the course tool). The AssessmentController class represents the main feature of the application, namely the creation of the peer assessment itself. Through the AssessmentController an Assessment is created and stored using the AssessmentDAO for the assessment to appear in the course content list. After an assessment is created the students submit their assessment answer handled by the SubmissionController which in turn creates a new Submission and store it using the SubmissionDAO. Once all submissions have been delivered the module decides who should review who's submission, this is done by the ReviewController which stores an StudentReview instance for each person who should review an submission. In the end the student selected to review an submission writes an evaluation of that submission, the evaluation is stored as an Evaluation in the database through the ReviewController using the EvaluationDAO.

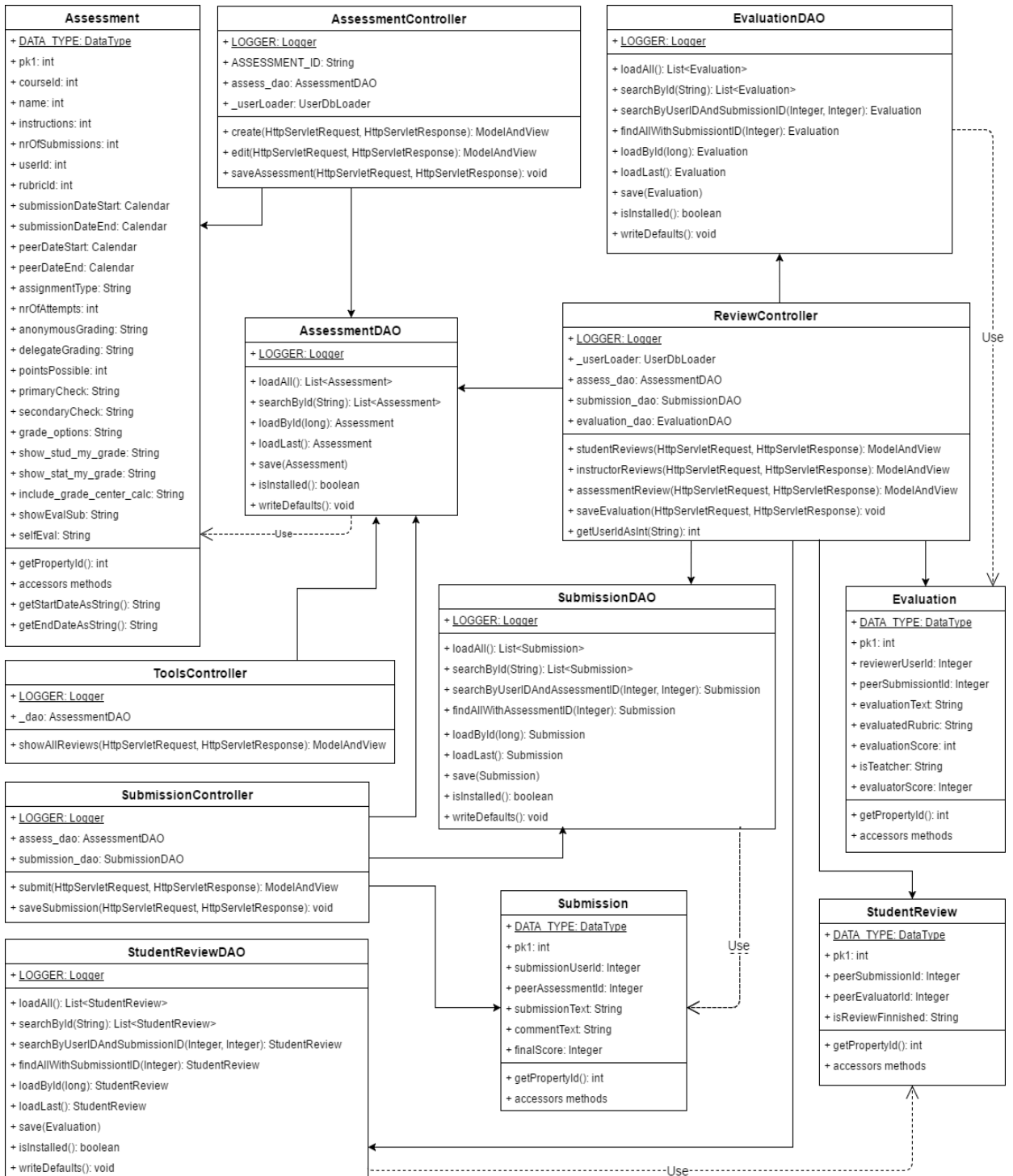


Figure 5.9: UML Class Diagram

5.5.3 Process view

The process view deals with the dynamic aspects of the system, it focuses on the run-time behavior of it by explaining how the different parts of the system are connected through communication. The process view addresses concurrency, distribution, integrators, performance, and scalability. In this paper, one large BPMN diagram represents the process of the system from the creation of a peer assessment to the delivery of an evaluation (Figure 5.10). The diagram displays the upper layer aspects of the system and gives an overview of the processes it goes through.

The process view starts with the creation of a new peer assessment, this directs the instructor to the creation page where they can add information about the assessment, set criteria (e.g like a rubric) and decide when the submission and evaluation dates should be. Once the process of creating an assessment is complete the assessment is stored in the database and added to the course content page.

Once the assessment is ready for student submissions the student will be able to view the assessment in the course content page. Students will then add submission data either as an uploaded file or as written text. When the assessment is submitted, the submission answer is stored in the database and the assessment is marked as completed.

The final step is the review step, where other students analyzes the submitted answers of their fellow students and creates a review. The review process includes reading the submission, assessing it based on the assessment criteria, and give a written feedback. The review is uploaded to the database and once the evaluation date has passed the review becomes available to the student who submitted the initial assignment.

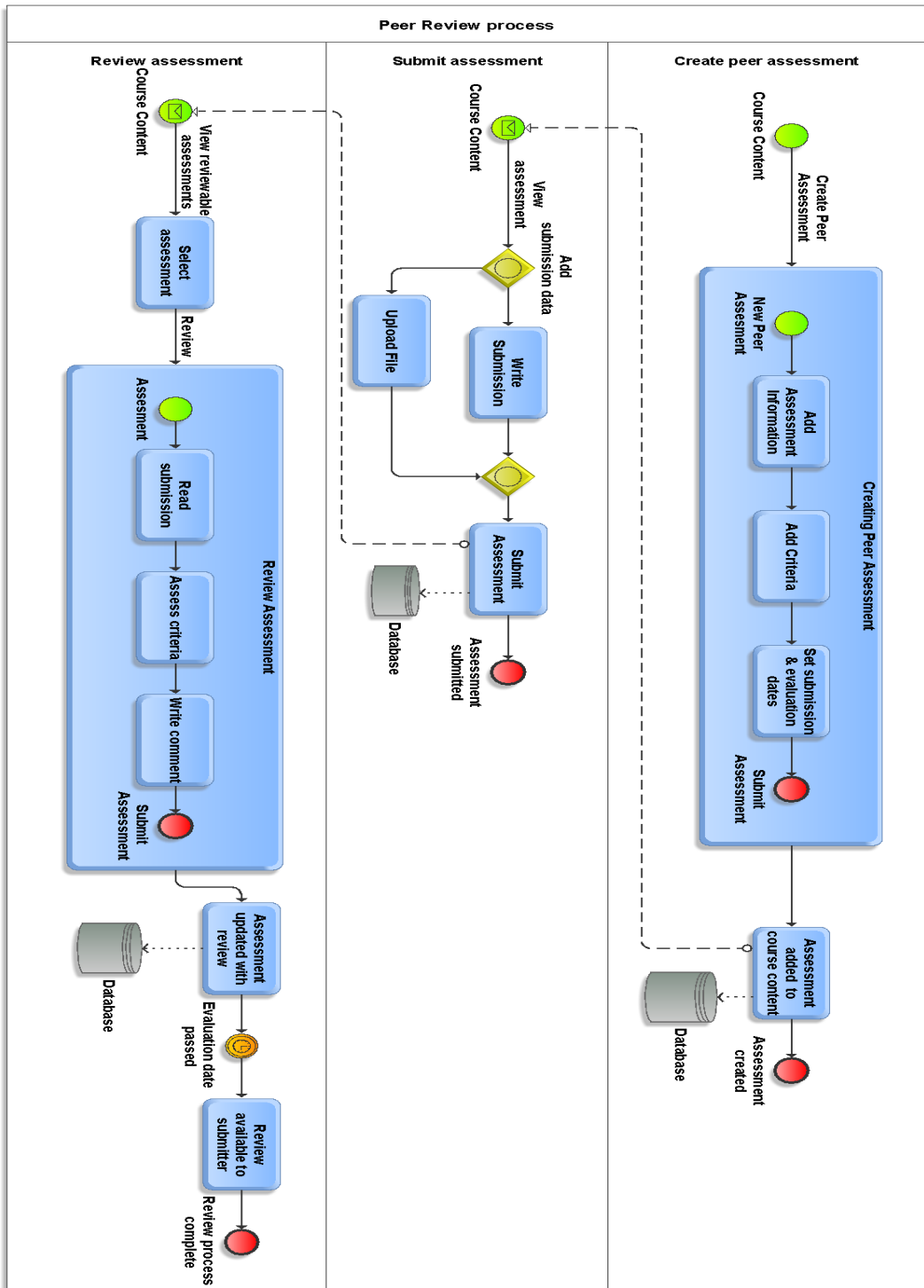


Figure 5.10: BPMN: Process view of the peer assessment creation

5.5.4 Physical view

The physical view depicts the system from a system engineer's point of view. It is concerned with the topology of software components on the physical layer as well as the physical connections between these components (Figure 5.11). The topology for our software is subject to change as it can be deployed on any system running the Blackboard learn software. The view represented here illustrates one of the topologies of a system running the peer assessment module. In this view, the student and instructor interact with Blackboard which in turn gives them access to the peer assessment module. Blackboard has its own internal API which the peer assessment module utilizes, and the whole software system is connected to a database containing the Blackboard schemas and tables.

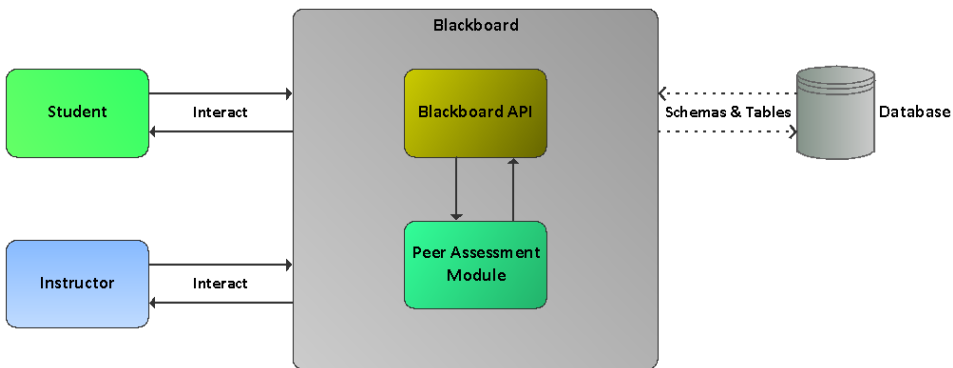


Figure 5.11: An illustration of the peer assessment module topology

5.5.5 Scenarios

The description of the architecture is illustrated using a small set of scenarios as use cases. The scenarios describe sequences of interactions between objects and processes. They are used to identify architectural elements and to illustrate and validate the architecture design. The first two figures display the first part of the peer assessment processes, namely the process of creating an assessment (Figure 5.12) and submitting an answer to that assessment (Figure 5.13). While the textual use case goes into details on the review process.

Assessment creation is done by the instructor who fills in the required and optional fields. The fields include: choosing when the submission and evaluation dates are, adding general information like name, and set the review criteria. Once the assessment is created (and given that the date is corresponding with the given submission date period) the student is notified and is required to submit their answer to the assignment. The student reads and understands the instructions, before they upload an answer file or write the answer directly into the text box available. The assignment is considered complete when the student has submitted an answer. The student can later look at the delivery status of the assignment in order to ensure its completion, and a completed assignment is required for it to be passed onto the review process.

In the textual use case "Review submission(s)" a student is required to review an submission provided by the peer assessment module. This process is completed as many times as there are submission to be reviewed, the number of submission to be reviewed is decided by the instructor when creating the assessment.

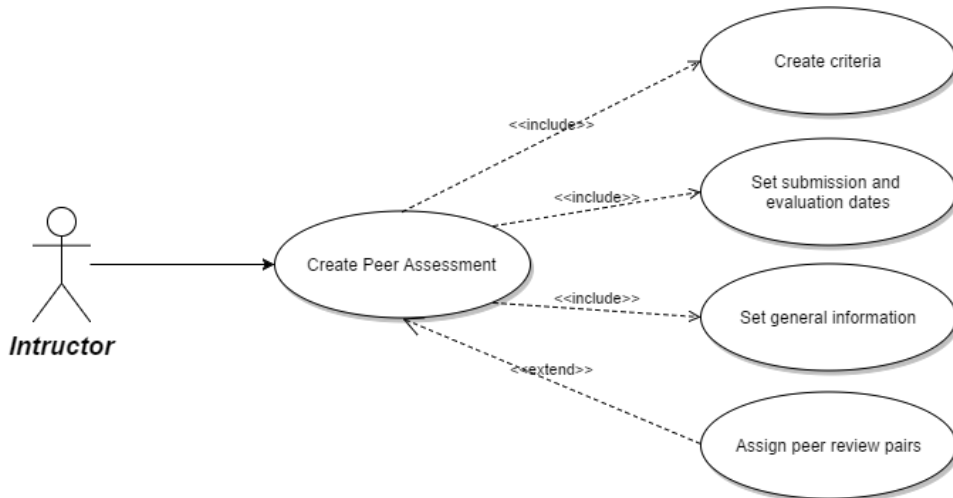


Figure 5.12: Use Case: Create peer assessment

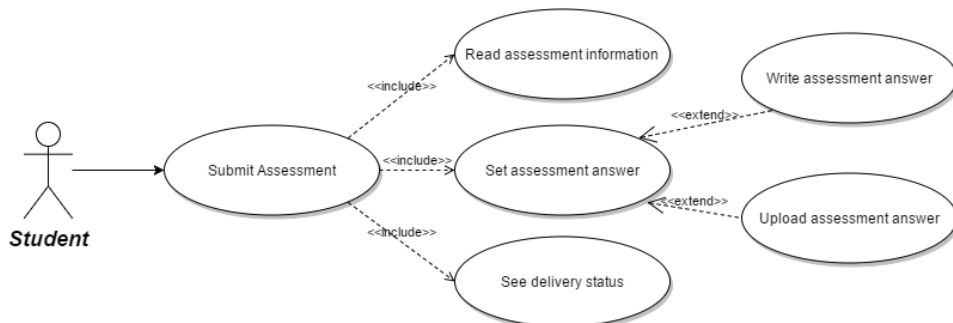


Figure 5.13: Use Case: Submit assessment answer

Use case name	Review submission(s)
Actors related	End-user: Student
Trigger	The user is prompted to review submission(s)
Pre-conditions	User is part of the course, submission date must be passed
Post-conditions	The review is submitted correctly and the user is taken to the assessment overview screen
Normal event flow	1) The user accesses the assignment 2) The user selects submission to review from the student reviews page list 3) The user reviews/evaluates the submission 3.1) The user writes a comment / note 3.2) The user sets criteria scores 3.3) The user uploads file(s) 4) The user submits the completed review
Variations of the event flow	3b) The user chooses not to do 3.2 or 3.3
Related information	The user should be able to do this for each review submission

Table 5.19: Textual Use Case: Review student(s) submission

Chapter 6

Development

In this project a digitized peer assessment module was developed for Blackboard. The module was built using Blackboards own building block framework and by utilizing theoretical and state of the art knowledge on peer review software. The level of Blackboard integration the building block had was more than most building blocks at the time, and the high level of integration required knowledge not widely available to the public. Blackboard advertise their LMS supporting third party development and this was something that was tested to the limit when implementing the peer review module.

The first part of this chapter is made with developers in mind as it goes into details on Blackboard development in general, and highlights some of the options available for developers. In the second part our prototype, its components, the implementation, and tools used in the process is elaborated. Lastly the testing done on our prototype is presented.

6.1 Blackboard development

When developing for Blackboard there are some key locations one should know about, in order to get support for a project. There are also several different implementation strategies one can choose from when implementing Blackboard modules. In this section the main sites for Blackboard support are listed and briefly explained. The central implementation strategies are also listed and described from a developers perspective.

6.1.1 Development support

Nothing is more important when starting to work on a development platform you are new to, than development support. It can sometimes be crucial to know where it is possible to find help if one is stuck in the development process or just need access to key platform documentation. There are multiple locations available to receive support for Blackboard development, ranging from discussion forums and API documentation to external sites containing previously made modules.

Blackboard Community

The Blackboard community site is the part of the Blackboard Learn site where developers share questions, insights and knowledge on developing for Blackboard [66]. Here it is possible to get help on questions regarding Blackboard development, but one can also find guides on how to setup certain Blackboard related things like building blocks or REST. Blackboard recommends their community site for questions regarding anything about Blackboard development.

Developer Documentation

The developer documentation for Blackboard is located inside the community site and contains all public available documentation on building blocks. This includes API documentation (e.g. Blackboard classes, their methods and variables), tag library (e.g. Blackboards own HTML tag library), and database schemas [67].

It is worth remembering that this is the documentation that is publicly available, not the documentation for everything in the Blackboard library. The nonpublic APIs can still be accessed and used in a building block through the Blackboard library (e.g. you can still create a Blackboard class that is not publicly documented), but there will be no documentation or support for this class. In addition to the documentation found in the community section there is also a section on REST on the Blackboard developer site [68].

External sites

The community site and API documentation offers help and explanation on Blackboard development, but it often directs you to external sites for larger code examples. These external sites are the best place too look if in need of examples of implementation or API usage, and some of the examples are also part of a fully working building block which can be tested in a Blackboard environment. One of these sites is OSCELOT [69], which contains a large number of independent Blackboard projects. The projects are everything from larger building blocks to small Blackboard fixes, most of which can be downloaded and launched inside the Blackboard Learn system. The site also contains documentation or source code for many of the projects, where developers who wishes to use similar (or the same) code can go and look at how a project used the API.

Another site that contains a lot of source code is Github [70]. An online hosting and version control service. Github contains many Blackboard building block examples, but finding the examples can sometimes be very difficult. Regardless of the difficulty finding the example code, Github remains one of the key locations for Blackboard building block code examples.

6.1.2 Implementation strategies

The choice of implementation strategy can be crucial for the available possibilities of a module, thus it is worth acquiring some knowledge on these before starting development. The strategies are separated by for instance integration options and external or internal application location, and the correct choice is heavily based on what possibilities you want for your module.

Building block

One strategy is Blackboard building block development, which can be used to extend or add functionality as well as integrate with external resources and services [71]. It is essentially an extension to the Blackboard software that can be developed by anyone who wish to add functionality to the Blackboard platform. The word "building block" highlights the fact that it is a block that can be added, removed and changed independently of the Blackboard system itself, but still being able to use and enhance the systems functionality. For those who wishes to build their own building block there is currently API documentation and a developer community available for supporting the development.

SOAP Web Services

If developers want to create modules or applications for Blackboard without implementing a building block, it is possible to do so with the SOAP Web Services. The Blackboard Learn SOAP Web Services are based on secure transmission between entities, such as the third party software and the Blackboard server. The transmission are using SOAP with WS_Security tokens. The Web Service itself is programming language-agnostic, making it very flexible for third-party developers to utilize.

REST API

The Blackboard REST API was the newest asset of development tools delivered by Blackboard when writing this thesis. It is a more modern approach of the SOAP Web Services, making it possible to do REST calls against an API. The REST API allows an outside source (e.g. a website or external service) to fetch and insert data from the Blackboard platform and utilize this externally. Currently, Blackboard has released a list of possible REST calls that are available and is continuing to build support for those who wish to start using this development strategy [68] (Figure 6.1). The REST API is the preferred approach if you already have an outside source, and it will be built upon more in the years to come to support more advanced REST calls and methods. In short, the main difference between building block and REST API development is that the former is a fully integrated software from the start, while the latter focuses on building an external web application, which is then integrated into Blackboard through tools like LTI.

courses			Show/Hide	List Operations	Expand Operations
GET	/learn/api/public/v1/courses	Get Courses			
POST	/learn/api/public/v1/courses	Create Course			
DELETE	/learn/api/public/v1/courses/{courseId}	Delete Course			
GET	/learn/api/public/v1/courses/{courseId}	Get Course			
PATCH	/learn/api/public/v1/courses/{courseId}	Update Course			
GET	/learn/api/public/v1/courses/{courseId}/children	Get Children			
GET	/learn/api/public/v1/courses/{courseId}/children/{childCourseId}	Get Child			

Figure 6.1: Part of the REST API for interacting with Blackboard courses

Learning Tools Interoperability (LTI)

Learning Tools Interoperability is a standard created by the IMS Global Learning Consortium [72]. This standards purpose is to connect learning systems such as LMS with external service tools. LTI tools are third party applications that are accessed from within the LMS (e.g. Virtual science experiments, interactive demons). In Blackboard this means that one creates a building block which essentially contains an outside application. With LTI, the developer can incorporate Blackboard building block API with an external application to get the best out of both strategies [73].

6.1.3 The Blackboard API

In the Blackboard API there are a number of accessible features and methods that enables the developer to get information on LMS specific content. Sometimes it is nice to know just what methods and features that are available when creating a building block for Blackboard. This subsection mentions a few that were used in this project which are relevant for all modules that revolves around making new assignment types.

Users and Courses

The API has classes with methods that can fetch information on all the users in the Blackboard database. This includes a users id, name, email, gender, and more. There are also similar classes for fetching course information, which means that the building block gets access to information about data like course id, enrollment dates, availability and title.

Grades and Grade Book

Most assignments involves some sort of final score or grade, this will also be true for most modules that digitizes those assignment types. Therefore it is valuable to know that Blackboard has methods that supports the enabling of grades for an assignment. It is also possible to set the actual grade from a different location then the default grade center by using these methods. Unfortunately the API documentation will not say anything about this functionality, but in Section 7.2.2 Figure 7.3 presents a code snippet containing the necessary code lines.

Entity access

Blackboard has its own database where all entities are stored, including students, courses, and much more. The database can be accessed by using explicit SQL queries, but the preferred method is through the API. In order to access a database entity through the API one need to instantiate the correct class, and run the desired fetch method on it. An example of this would be the *GroupDbLoader* class containing course groups. After the class is instantiated the method *loadByCourseId()* is ran with the id of the course, which in turn fetches all groups inside the course.

In Figure 6.2 below one can view the use of *GroupDbLoader* in the request-mapping method for assessment creation (e.g. the request ran when the user presses the "create a new Peer Assessment" link in Blackboard). The code shows how a ModelAndView object for the *create.jsp* is given the list of course groups, which in turn is used in the assessment creation page to display groups for group evaluation.

```
//Spring Autowire -- a.k.a. auto instantiate -- the GroupDbLoader object for use
@Autowired
private GroupDbLoader _groupLoader;

//Any time Spring receives a request for the URL with "create" inside assessment, this method is executed.
@RequestMapping("/assessment/create")
public ModelAndView create(HttpServletRequest request, HttpServletResponse response) {

    //The ModelAndView objects represents the jsp file with the given argument
    //E.g. in this case the jsp is called create.jsp
    ModelAndView mv = new ModelAndView("assessment/create");

    //Fetching the current course from context
    ContextManager contextManager = ContextManagerFactory.getInstance();
    Context ctx = contextManager.getContext();
    Course crs = ctx.getCourse();

    //Creates a list of all the groups in the course,
    //and adds them to the ModelAndView object as MultiSelectBean
    try {
        List<Group> groupList = _groupLoader.loadByCourseId(crs.getId());
        List<MultiSelectBean> multiSelectOptions = new ArrayList<MultiSelectBean>();

        //Converting a group into a MultiSelectBean
        //which is the class used in the <bbNG:multiSelect > tag
        for (Iterator iterator = groupList.iterator(); iterator.hasNext(); ) {
            Group group = (Group) iterator.next();
            MultiSelectBean selectOption1 = new MultiSelectBean(group.getTitle(),group.getId().toString());
            multiSelectOptions.add(selectOption1);
        }
        mv.addObject("groupsInCourse", multiSelectOptions);
    } catch (KeyNotFoundException e) {
        e.printStackTrace();
    } catch (PersistenceException e) {
        e.printStackTrace();
    }

    // Returning the ModelAndView tells spring to find the JSP and set these
    // objects in the context for the request.
    return mv;
}
```

Figure 6.2: Peer review module code showing the RequestMapping method that creates the ModelAndView object for create.jsp

6.2 Prototype

As a result of factors such as NTNU wanting to have full ownership of all their data as well as storing it internally, a close integration with the Blackboard platform was a high priority. Thus, with its solid integration and wide range of API options, the building block strategy was selected for the prototype development. The building block consisted of two main parts; namely the front-end written in JSP, HTML and Javascript; and the back-end which contained Java servlets, beans and database entities. This section starts out by describing what a developer should do before implementing their own building block, and what choices that were made in terms of creating our building block. It also documents how the front-end and back-end implementations were created, with code examples and views of the front-end taken from our peer review module.

The starting point for the development was Blackboard building block guides with implementation templates. The project began with one of these templates and was extended with its own contextual code and features.

6.2.1 Building block creation

When creating a building block there are a few things one need to consider before starting the implementation process. This consists of downloading the necessary software for development and finding the correct documentation for building blocks.

Developer Tools

To develop a Blackboard building block, there were a range of technologies and developer tools that where useful. Here, all of the most important ones are listed and described, to give a structured and simplistic overview.

- **STS - Spring Tool Suite**
Spring Tool Suite is an IDE based on Eclipse for developing Spring applications.
- **Spring Framework**
An application framework for developing Java application, and in this case, Java web applications.
- **Oracle VM VirtualBox**
A virtual machine container used for containing the Blackboard virtual machine that was developed against.
- **Gradle**
A build system for deploying building blocks onto the Blackboard virtual machine.
- **Git**
Version control for developing and distributing code in teams.

Preparations

The first and most important part of Blackboard development is the Blackboard virtual machine (VM) [74]. This virtual machine runs its own instance of the Blackboard Learn environment, thus enabling developers to quickly deploy, test and validate their building blocks through creating course instances, fictional students and instructors, all of which can be accomplished in the system admin view displayed in Figure 6.3. For developers who do not have access to other Blackboard test environments (e.g. Some institutions like NTNU have their own test instance of Blackboard Learn) the virtual machine is essential in order to run and test Blackboard building blocks. Unfortunately, the machine is only available to Blackboard partners. Note that it is possible to download and share the VM with everyone in the development team, but Blackboards licence agreement still applies for all parts involved.

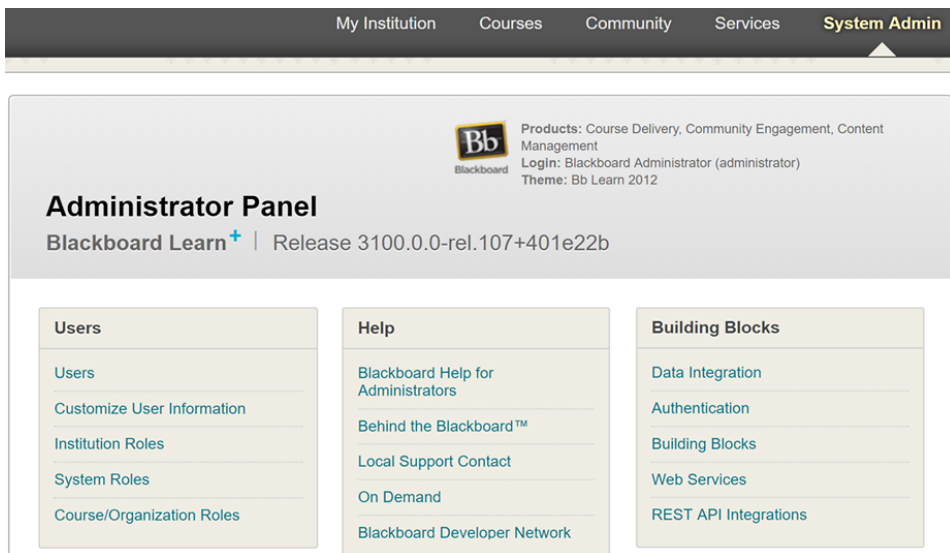


Figure 6.3: System admin page in Blackboard virtual machine environment

Once the Blackboard VM has been setup, the key to starting the building block development is to find the correct start-up documentation. Blackboard development can be difficult to figure out without any guidance, so using the documentation that is available is recommended. As mentioned earlier in Section 6.1.1, the documentation can be found at the Blackboard community site, and the best place to start is to read a specific set of posts on Blackboard building blocks start-up [75]. These posts includes guides on getting started with Blackboard building blocks, examples, and API documentation. Note that there are many other guides and examples on the forum that are not linked in these posts, which can provide additional information if necessary.

Creating the building block

After all the preparations are in place one can start the actual development. This section contains a short description of how the building block of this project was built, in addition to examples and documentation that was used to make it functional. The main reason behind the section is to provide guidance for others who are interested in setting up their own building blocks, and who wish to understand how this project was created.

For implementing the building block a software called Spring Tool Suite (STS) was used as the developer environment [76]. STS is an Eclipse based IDE which comes with the Spring framework built into it in addition to the existing Java and web support from Eclipse. At the time of development, the software was used by many building block developers, and STS suited our project perfectly since our building block was Spring based.

As stated earlier, the community documentation provides examples of building blocks, but also templates for starting building block development. The template used in this project was called "BBDN-Schema-Sample" [77] which was an example building block showing the usage of Spring for connection between database tables and the Blackboard table schema. This example block is ideal for usage in projects where one is creating entities that are to be stored and used in Blackboard's own internal database. The usage of Spring also allowed the module to use Spring annotation for actions like autowiring dependency injections and mapping requests to Java methods.

In order to have the building block running on the Blackboard VM, Gradle was used for building and deployment. Once the setup was complete (e.g. correct values are filled inside the build.gradle file) running and updating the building block was as simple as writing one line in the command line. It was also possible to download the log outputs from the building block which was mainly used for debugging purposes. The logs provided important information on both building block errors and database failures. To download the logs one had to log into the Blackboard VM as administrator and download the logs from the "tools and utilities" logs section.

Building issues

Note that this section only highlights some of the issues with the creation of our building block, for a full list of issues regarding the development itself see Section 7.3.

When setting up the building block there were a few issues that can be highlighted for future developers. These issues had simple solutions, but often were time consuming to figure out. One of these simple issues were Gradle dependencies, as they did not update themselves automatically and thus the IDE complained about lacking the correct libraries. In order to update the Gradle dependencies one had to manually press the project in STS and choose "Gradle STS" and "Refresh dependencies".

Another issue that was easy to fix was the dependencies related to using JSP. In the template project one would get an error saying that it was unable to find correct mapping to the

libraries required for JSP (specifically the `HttpServlet`). To fix this error a simple addition to the Gradle dependencies was necessary (Figure 6.4).

```
// define the project's dependencies
dependencies {
    providedCompile "javax.servlet:servlet-api:2.5",
                  "javax.servlet.jsp:jsp-api:2.1"
```

Figure 6.4: Code snippet required to enable use of `HttpServlet` in JSP files

A more detailed description of the different parts of the building block, including our development context through the use of examples, is presented in the next sections. The main focus in these covers the low-level implementation choices including front-end and back-end technologies.

6.2.2 Front-end

Front-end is the visualization of the data stored in the back-end database which also handles the interaction with the user, all of which is presented in the user-interface. The user-interface is displayed using JavaServer pages, consisting of JavaScript, HTML and Blackboards own library tags. This section covers the low-level implementation choices made in front-end (e.g. JavaServer Pages), in addition to displaying the direct results of the user-interface presented in Section 5.3.

JavaServer Pages (JSP)

The JavaServer Pages (JSP) represented the views in the MVC architecture. These displayed the different pages in the module and is translated into HTML tags by the server when deployed. JSP also makes use of standard HTML, CSS and JavaScript, but as it is a technology for developing dynamical web applications it also makes use more complex functionality such as for-loops, if and else.

Figure 6.5 shows the JSP for the student submissions page, where the students submit their initial assignment. While some of the code in the figure is not shown (for visualization purpose), one can still see how both if and for loops are used in "step1" to determine whether to display group names, and to loop through all available assessment files.

```
<bbNG:learningSystemPage ctxId="ctx">
  <bbNG:pageHeader>
    <bbNG:pageTitleBar title="{page_title} {assessment_name}"/>
  </bbNG:pageHeader>
  <!-- Checks if the submission is a group submission, and if the person is in a group or not -->
  <c:if test="{!isGroupSubmission || (isGroupSubmission && groupSubId != -1)}">
    <bbNG:form name="uploadForm" enctype="multipart/form-data" action="saveSubmission"
      method="POST" isSecure="{true}" nonceId="/submission/saveSubmission">
      . . .
    <bbNG:dataCollection>
      <!-- Step 1 Printing out assignment instructions, and link to files -->
      <bbNG:step id="step1_assignment_information" title="ASSIGNMENT INFORMATION">
        <h3 style="...">{assessment_name}</h3>
        <p>{assessment_instructions}</p>
        <c:if test="{isGroupAssessment}"><b style="...">Deliver as Group: </b> {groupName}<br></c:if>
        <c:if test="{assessment_files.size() > 0}" ><b style="...">Assessment Files: </b></c:if>
        <c:forEach items="{assessment_files}" var="file" varStatus="loop">
          <a href="downloadFile?...">{file.getName()}</a>
        </c:forEach>
      </bbNG:step>
      <bbNG:step id="step2_assignment_files" title="ASSIGNMENT FILES"> . . . </bbNG:step>
      <bbNG:step id="step3_assignment_text" title="ASSIGNMENT SUBMISSION TEXT"> . . . </bbNG:step>
      <bbNG:step id="step4_assignment_comment" title="ADD COMMENTS"> . . . </bbNG:step>

      <bbNG:stepSubmit showCancelButton="true" >
        <bbNG:stepSubmitButton label="{edit? 'Update' : 'Submit'}"/>
      </bbNG:stepSubmit>
    </bbNG:dataCollection>
  </bbNG:form>
</c:if>
<c:if test="{isGroupSubmission && groupSubId == -1}">
  This is a group submission Join an eligible group or have your instructor add your group to this assessment.
</c:if>
</bbNG:learningSystemPage>
```

Figure 6.5: Code snippet from the student_reviews.jsp used in the peer review module

The JSPs are connected through the use of the Spring servlets that creates ModelAndView Java objects that contains the dynamical information which are sent to the different views through a redirect method. While the underlying servlet and the dynamic nature of JSP helps make the website feel dynamic, one still need to use programming languages like JavaScript in order to do more complex operations.

JavaScript

The JSPs can, as in standard HTML documents, make use of JavaScript containing functionality for making the views dynamical when for instance clicking buttons. The main usage of JavaScript in this module was connected to the criteria rubrics, as methods regarding for instance adding or deleting scales was solved using scripts (Figure 6.6). Furthermore, the dynamical building of the rubrics in both the edit and assessment review page is accomplished using JavaScript together with JSP syntax on page load.

```
/* Deletes the most right scale of the rubric*/
function deleteScale(){
    var table = document.getElementById("rubric_table");
    var scale_header = document.getElementById("scale_header");
    var input = document.getElementById("numberOfScales");
    input.value = parseInt(input.value) - 1;
    scale_header.colSpan = scale_header.colSpan - 1;
    if(table.rows[1].cells.length === 2){
        alert("There have to be at least one scale");
        return;
    }
    for (i = 1; i < (table.rows.length - 1); i++) {
        table.rows[i].deleteCell(table.rows[i].cells.length - 1);
    }
    changeTotalPoints();
}
```

Figure 6.6: Code snippet from the create.jsp - function for deleting a criteria scale

Another example is when validating forms in the JSPs. There, one can create complex validation functions for checking more than empty input fields, which in turn are executed when trying to submitting the forms. These validation functions can be tailored to validate different variables or conditions. An example could be to ensure that submitted information is on the correct format, which would result in a higher interoperability between the components.

Covering the design aspects

The front-end tried to cover all the necessary aspects discovered from state of the art, theories and interviews. One of our main goals was to make the user interface as intuitive and easy to use as possible, thus this is something the final user-interface result should reflect. Below is a figure illustrating the connection between the different JSP views in the module, using the actual web pages in a scaled down format (Figure 6.7). The assessment evaluation view is presented in full scale later in this section, along with the reasoning behind its final design (for more full scale images see Appendix A).

The figure starts at the creation of an assessment, and shows how creating it enables the submission page, student reviews page and instructors evaluations page. After the submission is submitted the instructor can view the submission in the evaluations page and selected students will see the submission in their reviews page (which shows what submissions they need to evaluate). Both the instructor and student can, after receiving a submission, be directed to the evaluations page where they write an evaluation on that submission. Finally, once an evaluation is written, the student can look on the review of evaluation page to see the evaluations of their submission, conducted by other students, and review them back (if this option is specified by the instructor).

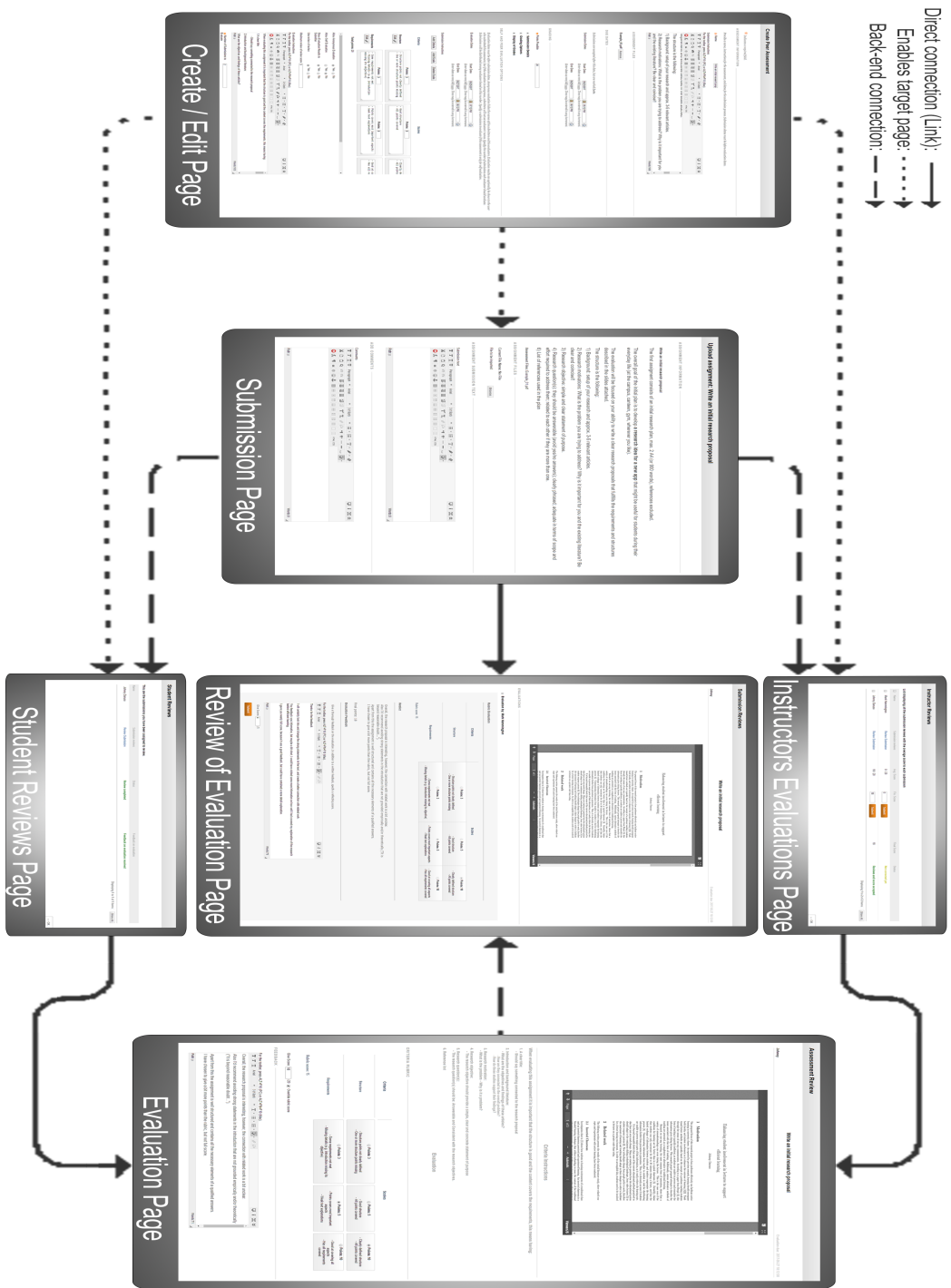


Figure 6.7: An overview displaying the different JSP pages used in the module and the connection between them (High Resolution Image, zoom is possible)

An example of a key functionality in the JSP that also were mentioned in Chapter 5 - Design, was the instructors ability to quickly display and finalize the score (grading) of a students submission. The design chapter describes a list of all submissions (with evaluations inside), where the instructor can simply press the submit button to submit the grade based on the average score given by the evaluators. For the implementation process, this meant getting direct access to the grade center in order to add and modify the submitted grade. The code required was not publicly documented, and an online meeting with an official Blackboard developer was necessary. The final result is visible in Figure 6.8, where the text on the right will become green when a grade is accepted and submitted by the instructor.

Instructor Reviews						
List displaying all the submission reviews with the average score to each submission						
Name	Submission reviews	Avg. Score	Fin. Score	Final Score	Status	
Mark Hammington	Review Submission	0 / 20	<input type="text" value="0"/> <input type="button" value="Submit"/>		Not corrected yet	
Johnny Daruso	Review Submission	18 / 20	<input type="text" value="18"/> <input type="button" value="Submit"/>	18	Reviews and score accepted	

Displaying 1 to 2 of 2 items [Show All](#)

Figure 6.8: A list over student submissions with evaluation scores ready to be given the final score

Another part of our user-interface, that also was mentioned in the design chapter, was the assessment evaluation process. The assessment evaluation page is where both the instructor and students write their evaluation of a submission. This page was the highlight of our chosen user-interface discussed in Section 5.3.3, and contained many of the key elements for making the peer review process as simple and streamlined as possible.

Below, in Figure 6.9, the final result of the assessment evaluation page is visualized, displaying a submission under evaluation. The important aspects from the design phase has been carried over, focusing on displaying all the elements (e.g. the submission, instructions and evaluation) on the same page in a structured way.

Some key features for the assessment evaluation page are:

- **PDF Viewer:** The PDF viewer from viewerJS allows the evaluator to easily read through the submission, while being able to write feedback at the same time.
- **Criteria Instructions:** The instructions from the instructor helps guide the evaluator.
- **Rubric:** By using the rubric the evaluator is guided into finding a balanced score, while also getting a better understanding of how much weight each criteria holds.
- **Custom Scoring:** Evaluators can choose to give score based on the rubric, but also has the choice to give a custom score.

Assessment Review

Write an initial research proposal

Johnny

Evaluation due: 2017-05-27 18:12:00



Criteria Instructions

When evaluating this assignment it is important that the structure is good and the content covers the requirements, this means having:

1. A clear title:
 - Should say something connected to the research proposal
2. Introduction and background literature:
 - What are the objectives and findings of these articles?
 - How are they connected to the overall problem?
 - How do these studies support their findings?
3. Research motivation:
 - What is the problem - Why is it a problem?
4. Research objective:
 - The research objective should provide a simple, clear and concrete statement of purpose
5. Research question(s):
 - The research question(s) should be: Answerable and Consistent with the research objectives.
6. Reference list

Evaluation

CRITERIA RUBRIC

Criteria	Scales		
Structure	<input type="radio"/> Points: 3 - Structure points not clearly defined - One or more structure points missing	<input type="radio"/> Points: 5 - Good structure - All points covered	<input checked="" type="radio"/> Points: 10 - Clearly defined structure - All points covered
Requirements	<input type="radio"/> Points: 3 - Some requirements not met - Missing details (e.g. introduction missing its objective)	<input type="radio"/> Points: 5 - Points covers most important aspects - Weak text explanations	<input checked="" type="radio"/> Points: 10 - Good at covering all aspects - Has all requirements covered

Rubric score: 15

Give Score: /20 Override rubric score

FEEDBACK

For the toolbar, press ALT+F10 (PC) or ALT+FN+F10 (Mac).

Overall, the research proposal is interesting, however, the connection with related work is a bit unclear. Also I'd recommend avoiding strong statements in the introduction that are not grounded empirically and/or theoretically ("It is beyond reasonable doubt...").

Apart from this the assignment is well structured and contains all the necessary elements of a qualified answers. I have chosen to give a bit more points than the rubric, but not full score.

Path: p

Words:71

Click Submit to finish. Click Cancel to quit without saving changes.

Cancel Submit

Figure 6.9: The final version of the assessment evaluation view (as seen by the evaluator)

6.2.3 Back-end

The back-end of the implementation consists of multiple database entities which are connected to their respective Java classes using Spring servlets (e.g. controllers), in addition to also having DAO classes and beans for extended support. An overview of how the MVC architecture is used with JSP and servlets is displayed in Figure 6.10.

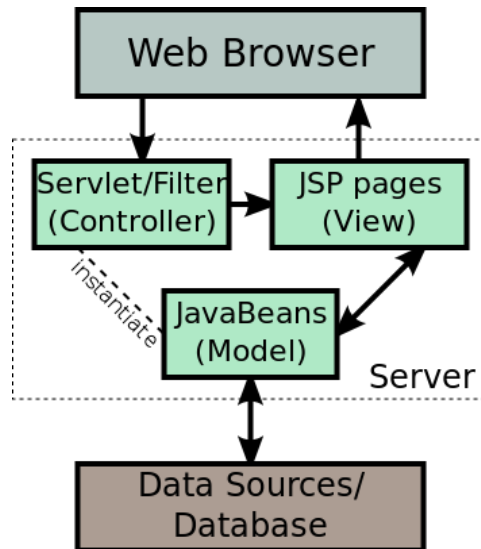


Figure 6.10: MVC spring architecture [78]

Spring Servlets

The main back-end entities that handles most of the logic and information flow are the controllers in the MVC architecture. In building block development, these controllers are more often referred to as servlets, handling rest requests. One can register each servlet in a file called *web.xml*, and create each one from scratch. Or it is possible to use other alternatives, such as the Spring Framework chosen in this project.

Spring is set up through a configuration file and only the Spring servlet is needed to be registered in the *web.xml* file. After that, one can create Spring controllers that makes use of request mappings for handling different types of requests and responses in the module. Also, it is the servlet methods that handles the dynamic data sent from the JSP views. They handle this data by verifying and manipulating it accordingly before storing them, through the use of data access object (DAO) Java classes, in the database. Most of the logic in the module is also placed in these methods, such as the algorithm for pairing up the students to evaluate each other.

Below in Figure 6.11 is an example of a controller used in the peer review module, namely the controller for the creation, editing and saving of an assessment. The content of the

controller in the image has been heavily pruned (as there are too many details to display in one figure), but it shows the three most important request methods used in the assessment page. These methods are similar to the methods used in the other controllers, and all of the methods are activated through Spring.

```

/**
 * Spring controller for handling all request mappings related to creating and editing an assessment.
 */
@Controller
public class AssessmentController {
    /** Autowire -- a.k.a. auto instantiate -- the DAO objects for use in the controller */

    /**
     * Method is ran when a new assessment is being created
     * (e.g. the instructor presses "Peer review" in course content, which opens a new window with an empty assessment).
     */
    @UserAuthorization("system.admin.VIEW")
    @RequestMapping("/assessment/create")
    public ModelAndView create(HttpServletRequest request, HttpServletResponse response) {
        ModelAndView mvCreate = new ModelAndView("assessment/create");
        /**
         * Creates a ModelAndViewObject used in the creation of an assessment, and fills it with the necessary default values.
         * This includes for example a list of groups for group assignments.
         */
        return mvCreate;
    }

    /**
     * Method is ran when the instructor presses "edit" on an already existing assignment.
     */
    @UserAuthorization("system.admin.VIEW")
    @RequestMapping("/assessment/edit")
    public ModelAndView edit(HttpServletRequest request, HttpServletResponse response,
        @RequestParam("parametre_1") String parametre_1
    ) {
        ModelAndView mvEdit = new ModelAndView("assessment/edit");
        /**
         * Creates a ModelAndViewObject used in the editing of an assessment.
         * The ModelAndViewObject is filled with all the values of the assessment (original name, score, information , etc)
         */
        return mvEdit;
    }

    /**
     * Method is ran when the instructor presses submit in either edit or create mode
     */
    @UserAuthorization("system.admin.VIEW")
    @RequestMapping("/assessment/saveAssessment")
    private void saveAssessment(HttpServletRequest request, HttpServletResponse response, RedirectAttributes rattrs,
        @RequestParam("parametre_1") String parametre_1,
        @RequestParam("parametre_2") String parametre_2
    ) {
        /**
         * Takes in all the arguments of the submitted form (e.g. the assessment field values like name, score, rubric),
         * and stores them inside the ntnu_peer_assessment database table, by using the Assessment class and AssessmentDAO.
         * Files added are stored in a separate folder, and if this assessment is a new assessment then it is added to course content.
         */
    }
}

```

Figure 6.11: Assessment Controller for the peer review module (Pruned to better visualize the Request Mapping Methods)

Beans

For the model part of the MVC architecture, one has the Beans. This is loose JavaBeans that represents different objects in the modules, ranging from simple to complex ones. These objects contains getters and setters methods which are used to change their contents. Often, these are the objects that the views, in this case the JSP files, are using to display the dynamic information in the module, and that are manipulated in the servlets.

In this case, each table in the database has its own Bean for manipulating and connecting the entities to the back-end servlets. For instance, the table *ntnu_peer_assessment* has the corresponding Bean called *Assessment.java*.

Data access object support (DAO)

The data access object (DAO) works as the link between the table entity Java class and the actual database table in Blackboard. When used, the DAO allows the Java table class to act as a class representation of the database entity, hence the variables are accessed and rewritten in the same way a normal object-oriented Java class would be. Once the variables has been rewritten the entire class is sent as an argument through the DAO where it is mapped to its respective database entity table inside the Blackboard database.

An example of the use of DAO would be the adding of information (e.g. name, instructions, score) to an Assessment, which is connected to a table entity Java class with the same name. The variables for name, points, etc, are set by a simple setter method(e.g. `assessment.setName(name)`), and afterwards the Assessment class is stored using an AssessmentDAO class. In the end, the DAO class runs an persistence connection to the Blackboard database and updates the database entity with the new assessment values.

Database

The database used for this project is the internal Blackboard database, meaning that all entities created by the peer review module are stored on the same server as the university courses and content. One clear advantage of this is that the content of the peer review module is always available (given that Blackboard itself is available), including both student submissions and evaluations.

The entities are created based on an XML schema which is linked to the modules manifest, and all new entities has to be defined in this schema. Below is a entity relationship (ER) diagram of the entities in the database that belongs to the peer review module (Figure 6.12). Most of the entities are self explanatory, like *ntnu_peer_assessment* which is the table entity for the assessment. The only table not directly linked to a specific view is the *ntnu_student_review* table, which is a table that specifies which submission a student should evaluate.

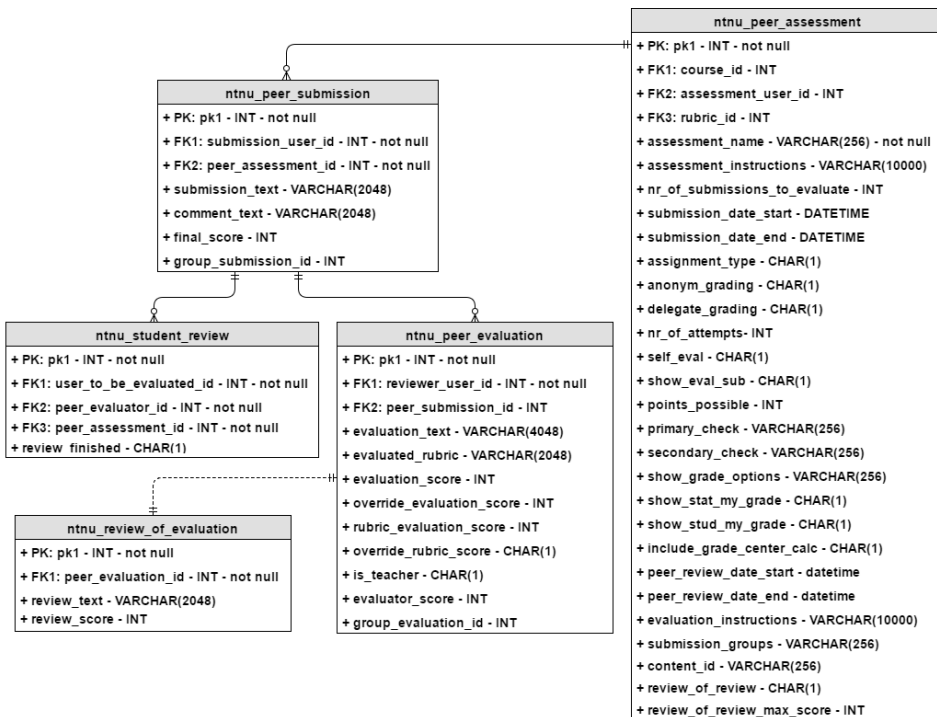


Figure 6.12: ER diagram over the peer review module database entities

6.2.4 Logic

Overall, the building block itself did not need to contain very much logic. Most of the work in implementing it was based on creating management control of different aspects such as view connection and database manipulation. Even so, there were some functionality that needed implemented logic.

Submission distribution

To distribute the submissions to be evaluated between the students, a pairing algorithm was needed. The algorithm focused on pairing up the students in a way such that each individual has to evaluate an equal amount of submission. In addition to this, it was also important to assign the evaluations in such a way that they got evenly distributed among the submissions. This, to ensure that each student got enough feedback on their submissions, in order to increase the overall quality of the evaluations. The reason behind this choice of implementation was based on some of the feedback gathered in the interviews [Section 4.2].

Rubric handling

In the already existing assignment and peer assessment building blocks, rubrics as guiding criteria for the instructors were already used. Unfortunately, it was not possible for external developers to access and make use of these rubrics. Thus, a new rubric solution was implemented. And as the already existing solution was a separate building block, it required a fair amount of code and logic to integrate it into our module. The main logic focused on creating the rubrics in the *assessment_review.jsp* on page load through the use of JavaScript, but also the storing and loading of database entities in a correct and structured way needed logical functions. This, in order to be able to extract and send the correct data to the JSP files, so that they in turn were able to build the rubrics correctly.

As mentioned in Section 5.4, the rubrics in the module is inspired by the ones in the already existing rubric building block in Blackboard, as well as the ones used in Canvas LMS. It emphasizes organizing the criteria simplistically, making it an effective way of defining the assessment evaluation guidelines.

6.3 Testing

To verify that the implemented functionality of the peer review module functioned as expected, it was important to conduct testing. In this section, the different types of testing conducted during this project is presented, including unit, integration and system testing. This is useful during development, to see that implemented functionality worked as intended, and if not, one would be able to correct it early.

6.3.1 Unit testing

Unit testing is a method where one tests small parts, or units, in a system, instead of testing the entire system every time a change is added. This is a clever way of testing, as it is easier to correct small units of a module than bigger parts, where the ripple effects might be larger.

Before implementing a new functionality in the module, a unit test scenario was defined to ensure that the implemented functionality was working as intended. These scenarios was described in a detailed fashion so that if there was a need to change some of the functionality at a later point, it was easy to re-test to see if the unit still functioned the way it should. An example of a unit test is given below in table 6.1, the rest is displayed in Appendix B.

	Description
ID	UNIT 1
Name	Required validation of fields when creating a new peer assessment
Date	April 25, 2017
Subject	create.jsp JavaScript validations
Steps	<p>In the create page for every required field:</p> <ol style="list-style-type: none"> 1. Fill in all the form data except the target required field and click submit. 2. Confirm that you are displayed a message requiring the field to be filled. 3. Fill in the required field and click submit again. 4. Confirm that the peer assessment is correctly submitted.
Results	Validation functioned correctly.

Table 6.1: Example unit test - required validation

6.3.2 Integration and system testing

After implementing a range of different types of functionality based on the requirements in Section 5.4, and testing these through the use of unit tests, one has to combine and integrate the different units into bigger components. After integrating all the components together, the complete system is created.

Even though the units might function in a correct way separately, it does not mean that they will behave like intended when integrated together. Thus, it was also important to create integration and system tests in the same fashion as with the unit tests, to guarantee that the interoperability between the units functioned correctly.

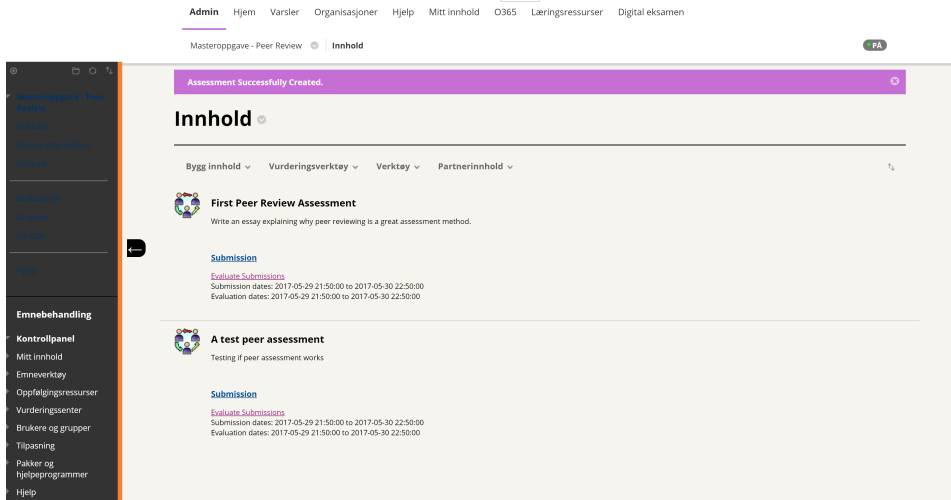
6.3.3 NTNU test server

As a final step of testing out the developed prototype it was deployed at the NTNU Blackboard test server. This, to see how it functioned in another environment than the virtual machine. In addition, it created a more real scenario of deployment as the building block had to be uploaded and added manually as a .war file, before enabling it in both the course tools and content for it to be ready for usage.

The key point in this step was to test out the default functionality of the module. This was accomplished by going through the use-cases specified in Section 5.5.5 and the complete BPMN process stated in Figure 5.10 in s Section 5.5.3.

There had to be done some small changes in the implementation for the module to function correctly in the new environment. For instance, the saving of database objects on the

test server was handled a bit differently as empty strings was interpreted as *null*, causing *NullPointerExceptions* to appear when attempting to load this information at a later stage. Thus, this showed the importance of testing out software in an environment more equal to the one it is to be deployed in, in order to fix small, but important bugs. A figure, displaying the module deployed on the test server is presented below (Figure 6.13).



The screenshot displays the NTNU test server interface. At the top, there is a navigation bar with links: Admin, Hjem, Varsler, Organisasjoner, Hjelp, Mitt innhold, O365, Læringsressurser, and Digital eksamen. Below this, the breadcrumb trail shows 'Masteroppgave - Peer Review' and 'Innhold'. A purple notification banner at the top of the main content area reads 'Assessment Successfully Created.' The main heading is 'Innhold'. Below the heading, there are several tabs: 'Bygg innhold', 'Vurderingsverktøy', 'Verktøy', and 'Partnerinnhold'. The 'Vurderingsverktøy' tab is active, showing two assessment items:

- First Peer Review Assessment**: Write an essay explaining why peer reviewing is a great assessment method.
 - Submission**: Evaluate Submissions. Submission dates: 2017-05-29 21:58:00 to 2017-05-30 22:50:00. Evaluation dates: 2017-05-29 21:50:00 to 2017-05-30 22:50:00.
- A test peer assessment**: Testing if peer assessment works.
 - Submission**: Evaluate Submissions. Submission dates: 2017-05-29 21:50:00 to 2017-05-30 22:50:00. Evaluation dates: 2017-05-29 21:50:00 to 2017-05-30 22:50:00.

A dark sidebar on the left contains navigation options under 'Emnebehandling' and 'Kontrollpanel', including 'Mitt innhold', 'Emneverktøy', 'Oppfølgingsressurser', 'Vurderingssenter', 'Brukere og grupper', 'Tilpassing', 'Pakker og hjelpeprogrammer', and 'Hjelp'.

Figure 6.13: Peer assessment successfully created on NTNU test server

Developer experience

Through our development process a lot of information and experience was gathered regarding the use of Blackboard as a development platform. This chapter is a summary of these experiences and it covers concerns like prerequisites required, platform limitations, and highlights challenges related to the development.

The main purpose of this chapter is to provide guidance for future developers, so that they may avoid similar pitfalls and challenges that was encountered in this project. The chapter highlights everything from bottlenecks, to advantages and disadvantages of our development choices.

7.1 Prerequisites

In any software development project there is usually a certain amount of prerequisites which is either required or recommended before starting the development process. When developing for the Blackboard platform the knowledge required beforehand depends on the implementation strategy (e.g. building block, REST API, from Section 6.1.2), and time scheduled for development. Therefore when choosing the implementation strategy one should be aware of the different skills required for each strategy.

In addition to the requirements based on the implementation strategy there is also a question of development processes and supported tools. The short answer to this however is that there is no direct tool requirement for developing to Blackboard, meaning that any IDE or version control like Github can be used. Section 6.2.1 gives an overview of what tools that were used in this project.

The building block strategy

In this project the building block strategy was chosen, which means creating a building block from scratch, with back end, front end, and logic. This choice requires either a larger

team of people which has good skills in specific software categories (e.g. front-end), or a smaller team where all members are full stack developers. The specific knowledge requirement goes from writing back-end using Java, SQL and XML, to front-end development using JSP, HTML, CSS and JavaScript. In addition to this one need much knowledge of the Blackboard API and tag library (the tag library can be optionally replaced with HTML in most situations), which can sometimes be poorly documented.

The REST strategy

The other viable option in comparison to the building block strategy is by using the REST API provided by Blackboard. When using REST one creates an external application which uses the REST API to access Blackboard data. By selecting this strategy the developers are free to use any development language and build strategy as long as it supports REST. This gives an additional layer of freedom compared to integrated strategies like building blocks. But the REST approach is not without issues, by selecting this approach the teams access to the Blackboards database are limited to what the REST API can provide, which is currently something Blackboard is working hard on improving. Knowledge required with the REST API approach is based on choices made by the development team (e.g. free choice of programming language), but since the entire application is made externally the team needs to build everything themselves which sometimes can mean more work than building blocks.

The LTI strategy

An alternative to using only REST API is to use LTI mixing building blocks with external applications. This would require the team to possess the knowledge required to create a simple building block, with the content of the building block made externally.

7.2 Blackboard as a development platform

This section explains some of the advantages, limitations and other experiences around the Blackboard platform as it was utilized in this project. As mentioned earlier, this project used the building block implementation strategy and focused on being fully integrated, therefore the experiences with fully integrated building blocks is the main focus here.

7.2.1 Full integration advantages

Full integration building block development presented itself with many challenges, but it also made it possible to do some things that external applications could not do.

The database used was Blackboards own database, thus our module could be designed to interact with all known and available Blackboard entities, and one did not have to create a database from scratch. It also made it possible to handle information on students, course, grades, and instructors in a secure and easy (e.g. access was available through the API)

way (Figure 7.1). For NTNU, this means that all the data on current students can be used without any further complications (e.g. no accessing of external database to get information), and one can therefore easily make modules that changes, updates or gathers student info.

```
//Spring Autowire -- a.k.a. auto instantiate -- the UserDbLoader object for use
@Autowired
private UserDbLoader _userLoader;
...

//Fetch the current course from context
ContextManager contextManager = ContextManagerFactory.getInstance();
Context ctx = contextManager.getContext();
Course crs = ctx.getCourse();

//Creates a list of all the users in the course and gets their email and name
List<User> userList = _userLoader.loadByCourseId(crs.getId());
for(User user: userList){
    String email = user.getEmailAddress();
    String name = user.getFamilyName();
}
...
```

Figure 7.1: Code snippet showing how our software accessed users and their information

Another advantage of having a fully integrated module was the use of tag library and API, meaning that instead of only using HTML when designing the user-interface, one had the possibility of using predefined Blackboard HTML tags. This meant that the looks and feel of the Blackboard platform could be achieved without too much hassle (e.g. no need for major changes in CSS file). The API had some of the same possibilities, which meant that the API methods could handle some of the standard Blackboard actions like storing a file or saving a grade. While some of these libraries are accessible without full integration (e.g. a building block connected to an external application using LTI), they are still listed as advantages of our approach.

7.2.2 Platform limitations

Blackboard as a development platform comes with many opportunities to enhance the LMS experience with custom features and self made modules, but there are also some limitations to what sort of features one can implement. The two main platform options that limits the possibilities of features that can be implemented are based on; whether the UI is dynamic or not and on the level of integration the module requires.

Non-dynamic front-end

Primarily the Blackboard front-end comes in two versions, the older and non-dynamic version, and the new dynamic version offered in the ultra course view which is included in the ultra experience version of Blackboard (Figure 7.5). This project will not go into

details on all the differences between the ultra experience dynamic version of Blackboard and the other versions, as the version used on NTNU and in this project is the non-dynamic version only. However there are some clear disadvantages by choosing the non-dynamic approach. The non-dynamic approach will among other things make it difficult for the front-end interface to update itself based on changes in information (e.g. when you press a button in Blackboard, which changes the current page information, the default action is to go to a new page instead of just updating the current page). This also means that newer frameworks like React and AngularJS are difficult to implement, but it is still possible to use languages like JQuery and JavaScript to achieve some dynamic features. Nevertheless, there are some reasons for not choosing the dynamic version, the main one being that it is still in an early stage of development and it also presupposes a different partnership deal than the non-dynamic version.

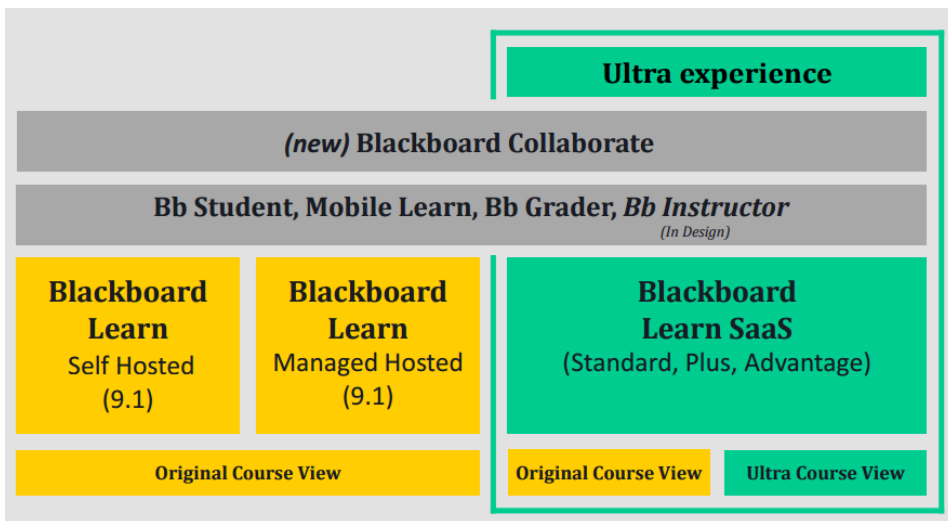


Figure 7.2: The different versions of Blackboard, where the Ultra Course View enables a more dynamic user interface. Currently NTNU uses the Manage Hosted 9.1 version

Back-end level of integration

While dynamic or static web pages are based on the version of Blackboard, the limitations in integration however are simply based on how reliant the module is on the Blackboard API. In short this means that there is a coherence between the the amount Blackboard features being used and the level of integration and dependency within the module. One could potentially create a module using only Blackboard framework and API, which would be limited to only the existing API and libraries of the blackboard platform, meaning that the module would simply be a combination of already existing features.

The module used in this project had a combination of using Blackboard API and external libraries, with the main focus of using the former. With the Blackboard API one could access users, grades and course content, thus being able to utilize important functional-

ity like updating the grades of students when the assignment was evaluated. Through the API and external library setup, one would also have access to database entities created by Blackboard and still be able to use HTML, JavaScript and CSS to present the data in an unique way. But there are also downsides to this approach, despite the fact that it seems like the best overall solution given that one can simply use the API whenever it is needed for back-end control and HTML on the rest. The downsides comes from the fact that the API is not very well documented, and sometimes the documentation is not even available. Looking at the Figure 7.3 given below where the grade of students are being set, gives an example of documentation which is not publicly available. The required methods are still available for developers to use, but for our project contacting official Blackboard developers was needed. They redirected us to a Github repository that contained the required code.

If your module do not require the usage of any special Blackboard API methods and is not supposed to be heavily integrated, then the approach for development can be quite different than it was for this project. Instead of using a lot of time on figuring out API methods and finding documentation one can simply go straight into development (given that one have basic knowledge of Java and other required languages). The only integration related issue will be to display the building block in an appropriate way, but since no other parts of the module requires Blackboard API one only need to spend time on the external requirements.

```

GradableItemManager giManager = GradebookManagerFactory.getGradableItemManager();
GradableItem column = giManager.getGradebookItemByContentId(contentId);

CourseMembershipDbLoader membershipLoader = null;
CourseMembership membership = null;

membershipLoader = CourseMembershipDbLoader.Default.getInstance();
membership = membershipLoader.loadByCourseAndUserId(courseId, userId);

//The DAO classes are autowired by spring
GradeDetailDAO gradeDetailDAO = grade_dao;
AssessmentDAO assessmentDAO = assess_dao;

// GradableItem column
GradeDetail userGrade = gradeDetailDAO.getGradeDetail(column.getId(), membership.getId());

//If no userGrade exists create a new one.
if (userGrade == null) {
    Id newUserGradeId = gradeDetailDAO.createNullGrade(membership.getId(), column.getId());
    userGrade = new GradeDetail();
    userGrade.setId(newUserGradeId);
    userGrade.setCourseUserId(membership.getId());
    userGrade.setGradableItem(column);
}

//Sets the score/grade
userGrade.setManualScore(Double.parseDouble(final_score));
userGrade.setManualGrade(final_score);

//Stores the grade in the database
gradeDetailDAO.persist(userGrade);

```

Figure 7.3: Code snippet showing how the peer review module sets the students grade

7.2.3 Deployment and testing

One of the most important things when developing is being able to test and run an actual instance of the product in a development environment. When developing for Blackboard, there are multiple ways to deploy and test your building block software. In this project Gradle and a virtual machine (VM) was used, in addition to NTNU's test server [6.2.1].

The peer review module was deployed onto the Blackboard VM using Gradle. By using the command line one could build, deploy and run the building block using only one command, *gradlew deployB2*. All build commands were defined in the *build.gradle* file provided with the building block project, and thus all build dependencies were also defined there. The use of Gradle made the deployment process simple, but it is worth noting that this still meant that one had to re-deploy every time a change was made.

The virtual machine is as explained in 6.2.1, a host for the Blackboard Learn environment. This meant that we had our own instance of Blackboard Learn, on which the building block module was deployed and ran. As a developer the ability to always have access to your own Blackboard instance is crucial, mostly because it allows you to test, modify and see the module in its real environment. The VM also allowed us to get log outputs on errors, debugging, and Blackboard access. The Blackboard VM worked very well for this project, apart from a few restarts and minor issues with the setup.

If all goes well with the deployment and testing on the VM, then the final step is to run the module on an actual test server. This is something that was conducted in the last part of the development process in order to test the module in a even more realistic environment. One could argue that the VM should provide a proper realistic environment, but the results of our testing detailed in Section 6.3.3 shows that this is sometimes only to a certain extent. In short, the test on the NTNU test server showed that multiple smaller adjustments had to be made in order for the peer review module to run properly. Our experience with the test server illustrated the importance of having such a step in the development process. By identifying small, but breaking bugs, one is able to ensure a higher overall quality of the module before reaching the production stage.

7.2.4 Blackboard support

In this project a lot of the development revolved around understanding the Blackboard API, documentation, and tag library. As mentioned earlier in Section 6.1.1, the developers have access to community forums, developer documentation and external sites. This section sheds some light on these support options and how they were perceived in this project.

Blackboard community

As mentioned earlier, the community consists of Blackboards own developers as well as other developers who work activity with Blackboard development. The feedback and help received from the community was a key component in solving some of the challenges faced in this project. There are many skilled developers who uses the forums as their main

platform for Blackboard development discussions, and some are very eager to help.

One can however not rely completely on the forum alone, despite the fact that it contains many skilled developers. The first reason is that many developers create completely different modules for Blackboard, thus your question might not be related to most of the other modules out there. In our case, the peer review module was more integrated than most modules, and thus there were a couple of questions regarding integration of some sort that did not find any answer on the forum. The second reason is closely related to the first, and it concerns the questions that required answers that only official Blackboard developers can provide. And because of this, the developer forums can sometimes not be enough, and reaching out to official Blackboard workers may be required.

Development documentation

The developer documentation consists of API, tag library and database documentation, all of which are available online through the Blackboard community website. In the development of the peer review module the documentation provided was used extensively, and it would have been impossible to make the module without it. All the regular tags, classes and entities are presented and documented, but the information provided with each differs from good to none.

Unfortunately, a range of classes (e.g. classes used by Blackboard themselves to access database, grades, etc) are not present in the documentation. This is something that has been mentioned earlier and is the reason for the majority of our development bottlenecks. On the positive side the classes are still available through the library provided by Blackboard, so it is still theoretically possible to use them (if one knows how).

External sites

The two main sites used and described in Section 6.1.1 were Github and OSCELOT. Github had a lot of projects that revolved around building blocks (in addition to many other Blackboard projects), and even some which were developed by the official Blackboard developers, containing code that was not part of the API documentation. These code examples were crucial for solving a few of the integration issues in our project, for example; the issue of setting a new grade.

OSCELOT also contains a large amount of existing Blackboard modules, which played a big part in giving examples of code usage. The code used in OSCELOT projects were used as references in our project in order to gain an understanding of how different components worked. Some OSCELOT projects even contained undocumented code as they had faced some of the same issues as this project. Below in Figure 7.4 one can see the OSCELOT project page where the projects are listed.

Full name	Account name (lowercase)	Description
YouTube mathup display fix	ymathupfix	The fix corrects the embed code to open YouTube videos instead of play with APT
File Resource locator	file-locator	This sys admin and course tool allows a user to look up a file in the Content Collection by searching on xID. It returns information about the file, including it's full path.
XCourseEmail	xcoursemail	This building block will create a course tool which will display email addresses and course id for all active courses for a specific instructor. This allows instructors to create email communication or collect addresses for all courses in a semester.
WordPress Integration Kit	wst-wordpress	Takes Blackboard users to a WordPress blog and logs them in automatically. The blog can also link to and embed content from the Blackboard content repository (a.k.a., file manager).
Who's Online	sen-whosonline	This Blackboard based building block displays Who's Online in a Blackboard system.
Welcome module for custom tabs	welcomemodule	The Welcome module is an add-on for the Community System which provides the "Welcome, {firstname}" header to any (custom) tab.
Webinar Integration Kit	webinar	Test project
WebCT ProxyToolbox	webctproxy	This PowerLink provides a framework within which separate tools can be made available to users via a single proxy tool instance. A tool can be any functionality; three tools are provided with the toolbox and users can write their own to be added. Do you have deployable components which need a permanent storage area for simple data values? The Proxy Tool Registry provides a simple utility to add this functionality to your tool.

Figure 7.4: The OSCELOT project page containing a large amount of Blackboard projects [69]

7.3 Challenges

As mentioned earlier, this project was under the experimental development paradigm. This meant that there were a lot of unforeseen challenges that could be encountered during the development of a building block at a new Blackboard institution. Below, the most significant challenges in this project are presented.

Not enough details in tutorials

At the start, when setting up the building block in the IDE, Spring Tool Suite, there were some templates and tutorials that was possible to use, but most of them was not covering enough of the setup seen in isolation. To be able to set up the building block completely with no prior knowledge, one has to make use of several tutorials as well as some trial and error.

Lack of a complete and thoroughly documented API

The biggest challenge encountered during the implementation of the building block was the lack of a complete and thorough documentation, both for the Java on the servlet side and for the JSPs on the view side. It is worth mentioning that there was still a good amount of documentation available on the Blackboard API, taglibrary (used in the JSP files) and database schemas, the problem was that there were a lot of inconsistencies in the quality of this documentation. For instance, the documentation on a tag from the tag library called *bbNG:contentList* stated that one had to set the *reorderUrl* attribute for the list to be drag and drop, but it did not give any examples. This problem was solved by looking at other tags in the library, as the *bbNG:hierarchyList* had the same attribute. The only difference was that in the documentation of this element, it was specified the possible values in the *reorderUrl* attribute, thus setting it to *noop* solved the problem.

Varying quality on community feedback

When not being able to find enough information in the documentation, the Blackboard community developers could potentially answer some of the questions. This is a really good thing, but as we created a building block that required a lot of integration, many

of the problems we encountered was not possible to get answers on from the standard community. Luckily, it was possible to schedule a meeting with one of the main developers of Blackboard, which knew almost all the answers to our development questions. The problem did not lie on the fact that the code was not there, but that some of the code was only documented in private APIs.

Difficult to correctly define the manifest file

Another challenge was related to setting up the *bb-manifest.xml* file correctly. The manifest file determines what permissions your module needs, and whether your building block will be displayed in places like course content, the course module list, or tools and utilities. The difficulty lied in figuring out what tags that was required to make the module show in the correct location, and to add the right permissions for the module. Our building block ended up using a tag called content-handler, which enabled the building block to be accessed from the course contents assessment option (Figure 7.5). For more information on what the different manifest tags mean, see the documentation found on All the Ducks website, a company focused on providing LMS development and integration services [79].

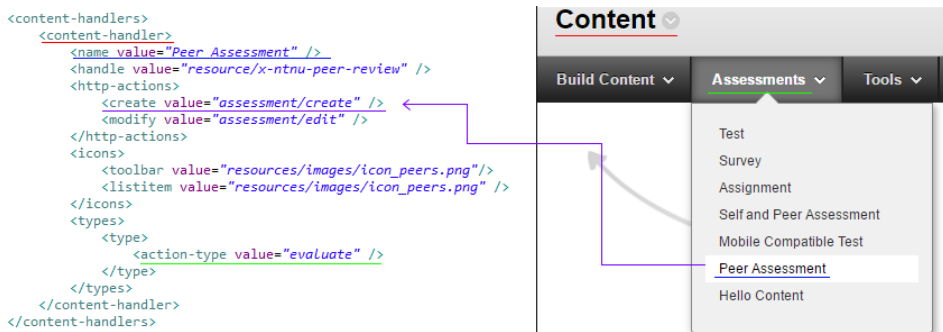


Figure 7.5: The correlation between the content-handler tag inside the manifest, and the appearance of a pressable link in Blackboard

Results and discussion

This thesis started off by researching on peer reviewing and the surrounding theoretical aspects, before designing and developing the peer review module and giving an overview of the different development experiences encountered during the implementation. In the upcoming section a summary of the key findings, based on the research questions, is given following an overview of the main contributions of this thesis.

Even though a functional prototype was developed, there were some functionality and other aspects that needed to be considered before being able to push the module into the production phase. These different aspects are listed in the future work section of this chapter, making it easier for future developers wanting to extend and finish the module.

8.1 Questions

The initial phases of this project focused on conducting research on peer reviewing as an assessment type, before developing the peer review module prototype as a part of an experimental development paradigm. During all of these phases there were some key findings which are presented in a detailed manner throughout the report. In this section, these findings are gathered together and summarized. Each of these are connected to the research questions stated in Section 1.1.

An underestimated assessment method

RQ1: How is the peer assessment method currently utilized in an educational setting?

Peer reviewing in a university context is still something that is not widely in use. It has been tested and made use of in a small subset of courses, but it is still not very wide spread as an evaluation method. From the interviews and the theoretical research, there is no doubt that this method, used in the right context, can have a huge learning outcome and success. Not only in the perspective of the evaluators, but also for the submitter of the

exercises, who receives feedback which is often can be more corresponding to their level of knowledge. Currently, peer reviewing is something that is successfully being used as an evaluation method in academic publishing. So why should it not be possible to accomplish the same results in a similar context, such as the university? Our research indicated that this is something which can be utilized even more in the future, given high quality preconditions, such as a peer review software and an proper explanatory introduction for the students.

Diversity

RQ2: Which features are important to include when implementing a digital peer review module

As a concept, peer review is connected to and can be utilized in several contexts, such as academic research and the university. As a results of this, peer review has a fair amount of diversity, thus a lot of choices were available when deciding on features and functionality before developing the software module.

Because of this, it is essential to conduct a thorough research and analysis, including gathering qualitative and/or quantitative data, when developing such a software. And through the research on theoretical aspects and interviews with potential end-users with relevant experience, important functionality was identified in this thesis. The most important ones were:

- **Guiding evaluation criteria:**
Criteria used as part of the evaluation process to enhance the feedback quality of the reviews.
- **Anonymous evaluation:**
Anonymizing the evaluators could be used for solving problems like arranging of score and feedback between students, or peers being harassed for giving critical feedback.
- **Group review:**
It is just as common and educational to be part of group exercises as individual ones.
- **Review of review:**
Reviews used in order to give feedback on evaluations received on submissions. This could be used to identify and analyze clusters of bad or good evaluations in order to improve in future exercises.
- **Automatic evaluation distribution:**
Automation of the evaluation distribution between the student submission was important, as this is a very time-consuming process if there are a lot of students attending to a course.
- **Organized assignment display:**
Organizing all the assignment content in such a way that both the submission and evaluation process is as convenient as possible.

It is important to note that the functionality identified in the peer review module in this project is not universal, and that development of similar software in other contexts would require a similar process of identifying features. Nevertheless, the classified functionality elaborated as part of this development is still something that should be taken into consideration. As the use of peer review in other contexts might have many similarities to how it is utilized in education.

Building blocks - cumbersome with a lot of potential

RQ3: What potential bottlenecks and concerns needs to be considered when developing a Blackboard module?

At the time of this thesis, it was fully possible to develop modules through the use of Blackboard as a development platform. Ranging from completely external ones making use of the REST API to fully integrated ones, such as the building block created in this project. Each specific development approach has its pros and cons, but the indications might suggest that development using the REST API is the future. This can be concluded by the fact that Blackboard themselves is promoting and refining the REST API continuously, in addition to the developer experiences [Chapter 7] in this project, giving the same indications.

Developing building blocks with Blackboard as a development platform per now is quite cumbersome. This, as a result of lacking overall quality documentation, important functionality only documented in private APIs, and relevant projects not easily found on Github. We find this really strange, as creating building blocks for Blackboard has a lot of potential, because of the possibilities to develop almost anything that needs to be fully integrated. As full integration is something that currently is much more difficult to accomplish with third party applications created with the REST API.

8.2 Contributions

Based on the different focus in the research questions, this thesis ended up with some distinct contributions for the future. Here, these are presented and reflected upon to present some concluding and summative knowledge for subsequent Blackboard development projects.

8.2.1 Module development

Through developing an integrated peer review module as a proof of concept on using Blackboard as development platform, it can be concluded that this is most certainly feasible for others as well. And as mentioned in the previous section, even though building block development can be inconvenient to some degree, the potential is there. So, if even more developers contributes through implementing Blackboard content and joins the Blackboard community, in addition to existing and private APIs getting refined and pub-

lished, the number of problems would decrease.

Nevertheless, it is safe to say that one can make use of Blackboard as a development platform in courses at NTNU or other universities. And one way to utilize this is through exercise projects aiming to let students develop their own building blocks. Still, it is worth pointing out that if one is to create software modules that does not require much integration, third party development with the REST API would probably be the better alternative.

8.2.2 Extended functionality

A lot of different functionality was specified and implemented for the peer review module. And even though it was an experimental development, some of the implemented functionality was included to extend beyond the functionality of the other peer review modules.

Especially, the use of review of review was something that had not been seen in many other peer review modules. Even so, this is a functionality that should be tested out, in order for the submission deliverers to be able to both defend their submission, but most importantly improve the initial evaluation level by giving feedback on it. This can be achieved through analyzing the feedbacks, both by the student given feedback to as well as the instructor to see if there are any patterns. Figure A.6 shows the part of review of review where the student gives feedback on the evaluation.

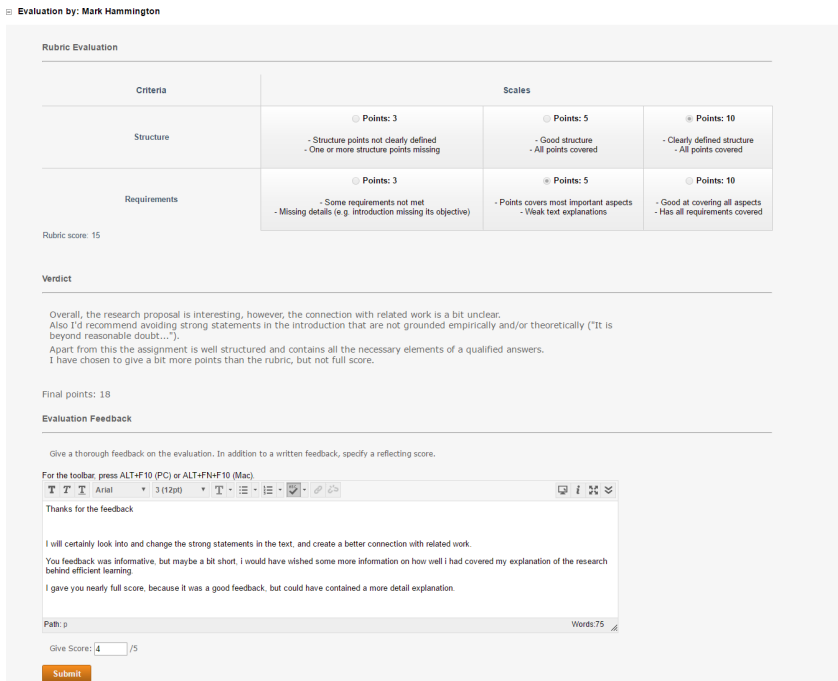


Figure 8.1: Part of the review of review page, displaying a student giving feedback on an evaluation

In addition to extending functionality, our module focused on increased usability. This was a direct reaction to the main problem with the already existing peer review module, which was its complexity. As displayed in Figure 5.4 in Section 5.3.3. Therefore the streamlining of the entire module was a high priority, in order for the usage of our module to be as seamless as possible. This could potentially lower the barrier for utilizing the new module by instructors, as a coherence with heightened usability and expanded functionality.

8.2.3 Blackboard Summary

After developing a building block for Blackboard it is clear that the platform has potential, and with the correct knowledge it can provide unique opportunities for the developer. The advantages can be access to university data, which makes it easier to create modules that manipulates course content or user information, or access to students, as it is mandatory for them to be a part of the LMS. These advantages are a big part of the reason why one would choose to develop for Blackboard directly and not make an external software only.

When developing for Blackboard a wide variety of skills are required depending on the implementation strategy. And for developers who wish to use their own choice of programming languages, the REST API would be the best alternative. But if one is going to develop small modules for a university course training program, a fixed building block setup would fit perfectly, as it would have direct integration, an already existing database, and not require any external connections.

The current main issue with Blackboard building block development is the lack of documentation and support. Blackboard provides general documentation on API and schema, but the API code that integrates Blackboard with university data is not well documented. It is also part of the Blackboard future to have a thriving community with support through other developers, but at the moment there are unfortunately many questions that the forum can not answer. This is however something that should improve over time with more developers joining the community. So, if one are to develop a highly integrated module one has to be prepared to contact some of the official Blackboard developers directly for guidance. This is not because the code is not existing, but because there is no documentation of it. The main reason for mentioning the lack of documentation is to ensure that future developers do not make the same mistakes as was made in this project. This included using a huge amount of time on finding the correct documentation and code examples for an integrated parts of the Blackboard and tag API.

To summarize, even if you are well prepared, developing your own module is still a lot of work. But if guided by a person of knowledge (e.g. a person with previous Blackboard development knowledge), the process will be much more convenient. A plausible scenario would be to have a course training program where the instructor is the person with development knowledge, and the students gets to create their own small modules for improving the course itself.

8.3 Project evaluation

In this project we started with a goal of discovering new information on peer review, making the peer review experience digital by integrating it fully with the Blackboard LMS. During the development process it became clear that Blackboard integration required much more focus than originally intended, and since Blackboard development was so important for NTNU we decided to shift our focus towards the development process instead.

We could of course had decided to digitize peer reviewing as an fully external module with a developer stack of our choice. Thus, being able to focus on trying to discover and innovate peer reviewing even more than during this project. The problem with an external module is that it would not necessarily been as useful or interesting for NTNU, and most likely not for other institutions neither. This, as most educational institutions wants to have ownership and storage of all the data used in such a module (e.g. student and instructor registration, as this is something that is handled automatically by a LMS).

The results of our choice gave us a deeper experience with Blackboard as a development platform, and still allowed us to get a better understanding of peer review as an assessment method. While we did not get to focus as much on peer reviewing as initially intended, it still made us interested in seeing what possibilities it could have in an education setting. We hope that in the future more instructors will use the peer review method, as it clearly can be rewarding for both students and instructor if used correctly.

Throughout our entire study period, we have been used to develop software with very popular and widely used frameworks and technologies, having access to large communities and thorough documentation. Thus, developing software without the usual amount of documentation, posed a challenge. Even so, the overall experience from this was in the end positive, as we learned a lot about developing without the necessary documentation (which is something that could occur when working as a developer). And as a result of this, it made us understand the value of having a network of competent people that could provide guidance when facing challenges.

While we would have liked to do more research on peer reviewing, the focus on Blackboard development has given us data and information that can be used in subsequent development projects. The result has already proven to be interesting for many who work with Blackboard at NTNU. And hopefully, this thesis will provide details that can be used as part of building competence around utilizing Blackboard as a development platform in the future.

8.4 Future Work

After developing a prototype of a peer review module for Blackboard, there were still some things that needed to be completed before the module could go into production. This section presents a list of aspects that needs to be considered before making use of the module in a real course setting.

8.4.1 Student data gathering

In this project, the focus has been on developing a peer review module with its core functionality, thus interviewing and gathering information around this was the main focus during the research phase. To improve on the module even more and to possibly find features for motivating the peer reviewers, it would be important to gather information from the students as well. This can be completed through the use of explicit information gathering methods such as interviews or questionnaires, but also methods like user and field testing. This, to map potential changes that needs to be done for the students to fully utilize the module.

8.4.2 Refined and extended functionality

Even though a lot of functionality was implemented as part of the prototype, there was still more functionality that could have been incorporated. Also, some of the implemented functionality itself still needed to be refined even more. For instance, the rubric implementation is working properly, but it should contain even more functionality for editing the rubric, like removing criteria rows.

From the state of the art, it was clear that the use of calibrated peer reviews was something that could have potential. It was not something that was a very common thing to do, but even so, it could be a good way to introduce the students to peer reviewing and the individual exercises. Thus, this was considered as something that could and should be added to the peer review module as extended functionality in the future.

Blackboard has their own mobile application (Figure 8.2) that can be used instead of logging into the LMS through the browser on the computer, but as the main focus in this project was on developing a building block to Blackboard itself, it was not prioritized to extend the mobile application as well. And so, this is something that needs to be further investigated, to look at the possibilities of extending the mobile application with the building block in order to make it useful and accessible there as well.

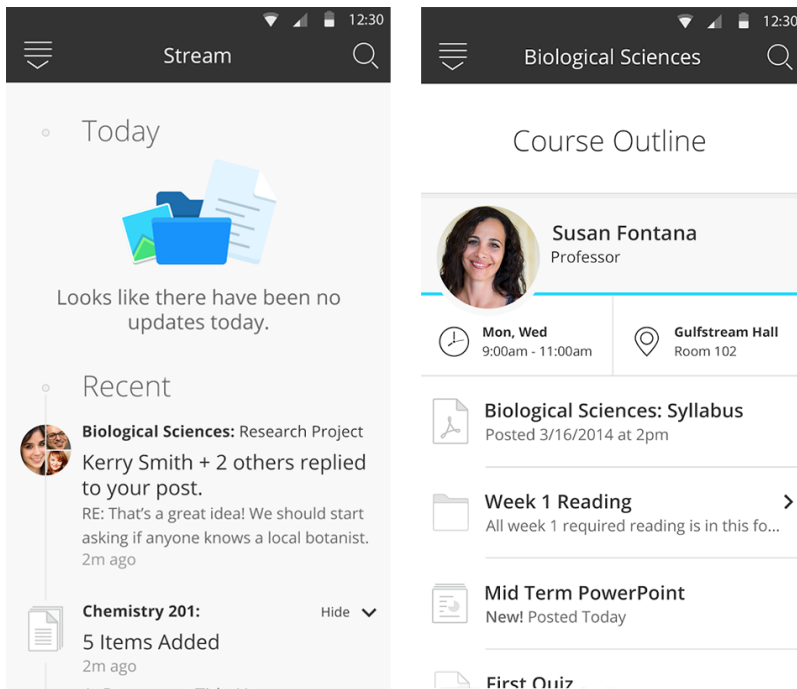


Figure 8.2: Blackboard student mobile application [80]

8.4.3 User interface

The developed peer review module was created based on the Blackboard theme through the use of the Blackboard tag library. As mentioned in Chapter 5 - Design, there are several things with the Blackboard user interface design that could be improved upon. This is something that should be reviewed by user interface or experience designers, as there might be small aspects of the module that can be changed to improve the overall quality.

8.4.4 Extensive testing

Before being able to push the module into production and actual use, it will be important to conduct even more extensive testing than completed during this thesis. Of course, one has to create unit and integration tests for new implemented functionality as well as running system tests. In addition to this and as mentioned in Section 8.4.1, some type of end user testing would be wise to conduct.

For instance, usability testing to gather qualitative test data through the use of the Wizard of Oz testing method [81] or eye tracking to exploit weaknesses in the software. To gather some more quantitative data, field testing could be a possibility, the module is then deployed and tested in a real setting over some time, where the users can give feedback.

Chapter 9

Conclusion

Developing building blocks for Blackboard did provide more challenges than initially expected. This, as a result of the module in development being much more integration dependent than many already existing building blocks, making many of the questions risen on the Blackboard community forum unanswered. The challenges did not arise as a consequence of lacking possibilities in the Blackboard development libraries, but because of the inadequate documentation of the API and tag library.

Our research has shown that both peer review and Blackboard development is something that NTNU should consider using more of, and that both the students and instructors could benefit from it. Within the next year Blackboard will be available on every NTNU students computer, and many courses could be using Blackboard development in their exercise program. This means that every student can make their own additions to Blackboard, thus the LMS at NTNU would not only improve by written feedback, but also from students adding their knowledge and suggestions directly into the LMS.

By researching on theoretical aspects surrounding peer review as a concept, as well as gathering qualitative data through interviewing NTNU professors, our research managed to extract relevant information and deeper understanding of how peer reviewing was used at NTNU. It was interesting to look into the different factors of peer review, and to develop functionality that let the student take the role of the instructor. If we had more time we would have loved to do some real life tests on actual students, to see if our module worked as intended. Nevertheless, our project was engaging, both in the sense that we gained a deeper understanding of peer review, and through our practical work of implementing our module.

Bibliography

- [1] Falchikov Nancy. *Improving Assessment Through Student Involvement*. Routledge-Falmer, 2005.
- [2] Keith J. Topping. *Peer assessment*. 2009.
- [3] Laura Guertin. Peer review. <https://serc.carleton.edu/sp/library/peerreview/index.html>. Last checked: February 15, 2017.
- [4] Teresa Berrow Robert Davies. *An evaluation of the use of computer supported peer review for developing higher-level skills*. 1998.
- [5] Charlotte West. *E-learning: The digitalization of swedish higher education*. 2007.
- [6] Blackboard Inc. About blackboard. <http://no.blackboard.com/sites/international/globalmaster/about/>. Last checked: February 02, 2017.
- [7] Briony J Oates. *Researching Information Systems and Computing*. SAGE Publications, 2006.
- [8] The University of Texas at Austin Faculty Innovation Center. *Methods of assessment*. <https://facultyinnovate.utexas.edu/teaching/check-learning/methods>. Last checked: February 14, 2017.
- [9] R.R. Reilly J. McGourt, P.Dominic. *Incorporating student peer review and feedback into the assessment process*. 1998.
- [10] Keith Topping. *Peer assessment between students in colleges and universities*. 1998.
- [11] Elsevier. *What is peer review?* <https://www.elsevier.com/reviewers/what-is-peer-review>. Last checked: February 15, 2017.
- [12] Sandra Tsui Eu Lam Guangwei Hu. *Issues of cultural appropriateness and pedagogical efficacy: exploring peer review in a second language writing class*. 2009.
- [13] Maria Ng Amy B.M Tsui. *Do secondary 12 writers benefit from peer comments?* 2000.

-
- [14] Randall W Sadler Jun Liu. The effect and affect of peer review in electronic versus traditional modes on 12 writing. 2003.
- [15] Karen Asenavage Ulla Connor. Peer response groups in esl writing classes: How much impact on revision? 2002.
- [16] Hui-Tzu Min. Training students to become successful peer reviewers. 2004.
- [17] The University of Texas Austin Faculty Innovation Center. Peer assessment. <https://facultyinnovate.utexas.edu/teaching/check-learning/feedback/peer-assessment>. Last checked: February 17, 2017.
- [18] J M Ribera et al. E Cobo, J Cortés. Effect of using reporting guidelines during peer review on quality of final manuscripts submitted to a biomedical journal: masked randomised trial. 2011.
- [19] Kevin Reiling Paul Orsmond, Stephen Merry. The importance of marking criteria in the use of peer assessment. 2006.
- [20] Vicki Byard. Power play: The use and abuse of power relationships in peer critiquing. 1989.
- [21] Lorraine A.J. Stefani. Peer, self and tutor assessment: Relative reliabilities. 2006.
- [22] Gonca Yangin Eksi. Peer review versus teacher feedback in process writing: How effective? 2012.
- [23] Linda Bol Ruiling Lu. A comparison of anonymous versus identifiable e-peer review on college student writing performance and the extent of critical feedback. 2007.
- [24] Falchikov Nancy. *Learning Together Peer Tutoring in Higher Education*. Routledge-Falmer, 2001.
- [25] Boundless Education. What is pedagogy? <https://www.boundless.com/education/textbooks/boundless-education-textbook/curriculum-and-instructional-design-3/instructional-design-14/what-is-pedagogy-48-12978/>. Last checked: February 03, 2017.
- [26] Edutech. Pedagogic strategy. http://edutechwiki.unige.ch/en/Pedagogic_strategy. Last checked: February 03, 2017.
- [27] Caroley Ames. Classrooms: Goals, structures, and student motivation. 1993.
- [28] Corrine Laverty. Pedagogic theory. <http://www.informationliteracy.org.uk/teaching/developing-your-teaching/pedagogic-theory/#squelch-taas-tab-content-0-1>. Last checked: February 03, 2017.

-
- [29] University of Michigan Center for Research on Learning and Teaching. Active learning. <http://www.crlt.umich.edu/tstrategies/tsal>. Last checked: February 13, 2017.
- [30] Jean Underwood Roberto Carneiro, Karl Steffens. Self-regulated learning in technology enhanced learning environments. 2005.
- [31] I. Chen. *Chapter 1.8: Instructional design methodologies. In: Instructional Design: Concepts, Methodologies, Tools, and Applications Information Resources Management Association, USA*. Information Science Reference, 2011.
- [32] Chico State Instructional Design & Technology Society. Addie model. <http://www.csuchico.edu/idts/addie.php>. Last checked: February 03, 2017.
- [33] Edutech. Educational technology. http://edutechwiki.unige.ch/en/Educational_technology. Last checked: February 03, 2017.
- [34] Kursat Cagiltay Hatice Sancar. Effective use of lms: Pedagogy through the technology. 2008.
- [35] The Moodle Project. Moodle. <https://moodle.org/>. Last checked: February 2, 2017.
- [36] Harvey Mellar Jose Enrique Hinostroza. Pedagogy embedded in educational software design. 2001.
- [37] Peggy A. Ertmer and Timothy J. Newby. Article update: Behaviorism, cognitivism, and constructivism: Connecting 'yesterday's' theories to today's contexts. 2013.
- [38] Intercollege Nicosia Cyprus Charalambos Vrasidas. Issues of pedagogy and design in e-learning systems. 2004.
- [39] Williams Caroline C Williams, Kaylene C. Five key ingredients for improving student motivation. 2011.
- [40] Tricomi Elizabeth DePasque, Samantha. Effects of intrinsic motivation on feedback processing during learning. 2015.
- [41] Richard M. Ryan and Edward L. Deci. Intrinsic and extrinsic motivations classic definitions and new directions. 2000.
- [42] Chin-Chung Tsai Sheng-Chau Tseng. Taiwan college students' self-efficacy and motivation of learning in online peer assessment environments. 2010.
- [43] Allan Wigfield Jacquelynne S. Eccles. Motivational beliefs, values, and goals. 2002.
- [44] Karyll N. Saari Lise M. Latham Gary P. Locke, Edwin A. Shaw. Goal setting and task performance: 1969–1980. 1981.
- [45] Erin Schoenfelder Tim Urdan. Classroom effects on student motivation: Goal structures, social relationships, and competence beliefs. 2006.

-
- [46] Learning management system. https://en.wikipedia.org/wiki/Learning_management_system. Last checked: February 02, 2017.
- [47] Itslearning - educational learning management system. <https://en.wikipedia.org/wiki/Itslearning>. Last checked: February 23, 2017.
- [48] Blackboard Inc. Blackboard peer review. <https://www.youtube.com/watch?v=Pv3cDy9gIp0>. Last checked: March 01, 2017.
- [49] CanvasLMS. About canvas. <https://www.canvaslms.com/about-us/>. Last checked: February 02, 2017.
- [50] University of Washington. Canvas peer review. <https://www.uwb.edu/learningtech/canvas/canvas-for-students/peer-review>. Last checked: March 01, 2017.
- [51] Google classroom. <https://support.google.com/edu/classroom/answer/6020279?hl=en>. Last checked: February 23, 2017.
- [52] Alice Keeler. Rubric template. <http://alicekeeler.com/2015/06/14/google-classroom-using-rubric-tab-to-assess-students/>. Last checked: February 03, 2017.
- [53] Turnitin. Peermark. https://guides.turnitin.com/01_Manuals_and_Guides/Instructor_Guides/Turnitin_Classic_for_Instructors/23_PeerMark. Last checked: February 23, 2017.
- [54] Turnitin. <http://turnitin.com/>. Last checked: February 23, 2017.
- [55] Peermark peer review example. <https://www.youtube.com/watch?v=zbomSYfWUIQ>. Last checked: March 01, 2017.
- [56] University of Melbourne. Praze. <http://peerreview.cis.unimelb.edu.au/tools/about-praze/>. Last checked: February 23, 2017.
- [57] University of California. Cpr - calibrated peer review. <http://cpr.molsci.ucla.edu/Overview.aspx>. Last checked: February 23, 2017.
- [58] Michael Lee Wesch. Cpr - peer review. [http://ksuanth.wikifoundry.com/page/How+calibrated+peer+review+\(CPR\)+works](http://ksuanth.wikifoundry.com/page/How+calibrated+peer+review+(CPR)+works). Last checked: March 01, 2017.
- [59] Stack overflow. <https://stackoverflow.com/>. Last checked: April 1, 2017.
- [60] Studyblue flashcards & quizzes. <https://play.google.com/store/apps/details?id=com.studyblue&hl=en>. Last checked: May 28, 2017.
- [61] University of Washington. Shneiderman's eight golden rules of interface design. <https://faculty.washington.edu/jtenenbg/courses/360/f04/sessions/schneidermanGoldenRules.html>. Last checked: March 02, 2017.

-
- [62] Euphemia Wong. User interface design guidelines: 10 rules of thumb. <https://www.interaction-design.org/literature/article/user-interface-design-guidelines-10-rules-of-thumb>. Last checked: March 27, 2017.
- [63] Tom Ewer. 10 rules of good ui design to follow on every web design project. <https://www.elegantthemes.com/blog/resources/10-rules-of-good-ui-design-to-follow-on-every-web-design-project>. Last checked: March 27, 2017.
- [64] Viewerjs. <http://viewerjs.org/>. Last checked: March 27, 2017.
- [65] Philippe Kruchten. Architectural blueprints—the 4+1 view model of software architecture. 1995. Last checked: February 24, 2017.
- [66] Blackboard community. <https://community.blackboard.com/community/developers/learn>. Last checked: April 08, 2017.
- [67] Scott Hurrey. Blackboard api documentation. <https://community.blackboard.com/docs/DOC-1115-building-block-api-documentation>. Last checked: April 11, 2017.
- [68] Blackboard Inc. Blackboard rest. <https://developer.blackboard.com/>. Last checked: March 28, 2017.
- [69] Paul Erickson. Oselot. <http://projects.oscelot.org/gf/>. Last checked: April 14, 2017.
- [70] Git hub. <https://github.com/>. Last checked: April 14, 2017.
- [71] Scott Hurrey. Blackboard building blocks. <https://community.blackboard.com/docs/DOC-1111>. Last checked: February 17, 2017.
- [72] Blackboard Inc. Lti blackboard help. [https://help.blackboard.com/Blackboard_Open_Content/Administrator/Open_Content_Technical_Details/Learning_Tools_Interoperability_\(LTI\)](https://help.blackboard.com/Blackboard_Open_Content/Administrator/Open_Content_Technical_Details/Learning_Tools_Interoperability_(LTI)). Last checked: April 14, 2017.
- [73] Blackboard Inc. Lti blackboard. https://help.blackboard.com/Learn/Administrator/SaaS/Integrations/Learning_Tools_Interoperability. Last checked: April 14, 2017.
- [74] Blackboard Inc. The blackboard virtual machine. <https://community.blackboard.com/docs/DOC-1104>. Last checked: May 03, 2017.
- [75] Blackboard Inc. Community: Blackboard building blocks. <https://community.blackboard.com/docs/DOC-1111>. Last checked: May 03, 2017.

-
- [76] Spring. Spring tool suite. <https://spring.io/tools>. Last checked: May 03, 2017.
- [77] Blackboard Inc. Bbdn-schema-sample. <https://github.com/blackboard/BBDN-Schema-Sample>. Last checked: May 03, 2017.
- [78] Wikipedia. The jsp model architecture. https://en.wikipedia.org/wiki/JavaServer_Pages#/media/File:JSP_Model_2.svg. Last checked: May 03, 2017.
- [79] All the ducks. Blackboard reference documentation. <https://docs.alltheducks.com/>. Last checked: May 04, 2017.
- [80] Blackboard Inc. Bb student app. <https://play.google.com/store/apps/details?id=com.blackboard.android.bbstudent&hl=no>. Last checked: May 14, 2017.
- [81] Timothy Griffin. Wizard of oz. <http://www.usabilitynet.org/tools/wizard.htm>. Last checked: May 07, 2017.

Appendices

Appendix A

Peer review module pages

Following is some of the views/pages that were not shown in full scale, or at all, in the rapport. These views shows how the final prototype looked like, while running on our internal Virtual Machine. The views are presented in order of appearance.

Creating a new Peer review assessment

Figure A.1 shows how the instructor starts creating a new peer review assessment by going to course content and pressing "Peer Assessment" in the Assessments drop down menu.

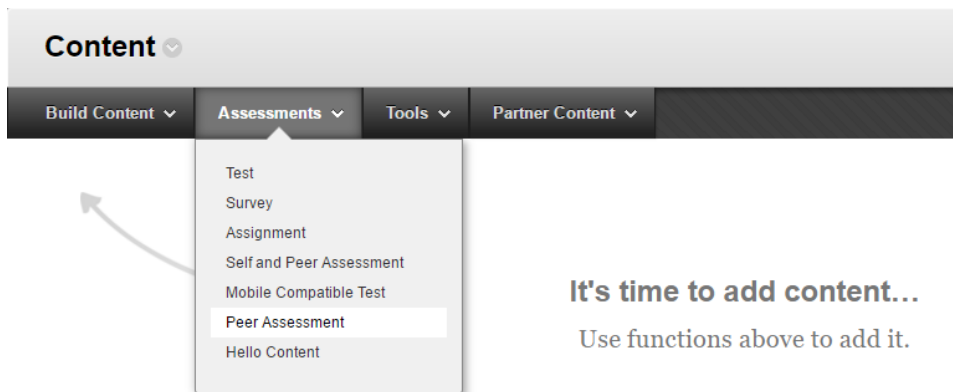


Figure A.1: Pressing the peer assessment option in course content

Filling inn assessment information

Figure A.2 is a full size version of the assessment creation page, were the instructor fills in the information (e.g. name, criteria, score) for the assessment.

Create Peer Assessment

✳ Indicates a required field.

ASSESSMENT INFORMATION

Provide a name, instructions for the assessment, and dates for the submission process. Submission dates must be before evaluation dates.

✳ Name

Submission Instructions

For the toolbar, press ALT+F10 (PC) or ALT+FN+F10 (Mac).

requirements and structures described in the slides attached.

The structure is the following:

- 1) Background: setup of your research and approx. 3-5 relevant articles.
- 2) Research motivations: What is the problem you are trying to address? Why is it important for you and the existing literature? Be clear and concise!

Path: p Words:169

ASSIGNMENT FILES

Example_01.pdf

DUE DATES

Submissions are accepted after this date, but are marked Late.

Submission Dates

Start Date

Enter dates as mm/dd/yyyy. Time may be entered in any increment.

End Date

Enter dates as mm/dd/yyyy. Time may be entered in any increment.

GRADING

✳ Points Possible

- Submission Details
- Grading Options
- Display of Grades

SELF AND PEER EVALUATION OPTIONS

Evaluation dates must be after submission dates. Anonymous evaluation hides the names of the submitters and the evaluators. Evaluation results can optionally be shown to the user who submitted the assessment, but if the evaluation is anonymous, submitters will not see evaluators' names. Specify the number of submissions each evaluator should evaluate. Submissions will be distributed among evaluators based on this number. Specify 0 submissions to evaluate if this assessment is only for self evaluation.

Evaluation Dates

Start Date

Enter dates as mm/dd/yyyy. Time may be entered in any increment.

End Date

Enter dates as mm/dd/yyyy. Time may be entered in any increment.

Submission Instructions

Criteria	Scales	
Structure <input type="button" value="Edit"/>	Points: 3 - Structure points not clearly defined - One or more structure points missing	Points: 5 - Good structure - All points covered
Requirements <input type="button" value="Edit"/>	Points: 3 - Some requirements not met - Missing details (e.g. introduction missing its objective)	Points: 5 - Points covers most important aspects - Weak text explanations

Total points: 20

Allow Anonymous Evaluation Yes No

Allow Self Evaluation Yes No

Show Evaluation Results to Submitter Yes No

Use review of review Yes No

Maximum review of review score

Evaluation Instructions

For the toolbar, press ALT+F10 (PC) or ALT+FN+F10 (Mac).

When evaluating this assignment it is important that the structure is good and the content covers the requirements, this means having:

1. A clear title:
 - Should say something connected to the research proposal
2. Introduction and background literature:
 - What are the objectives and findings of these articles?

Path: p Words:103

✳ Number of Submissions to Evaluate

Click **Submit** to finish. Click **Cancel** to quit without saving changes.

Figure A.2: The assessment "Write an initial research proposal" as shown inside the course content list

Course content list

Figure A.3 shows how an assessment looks when appearing inside the course content list, after its initial creation.

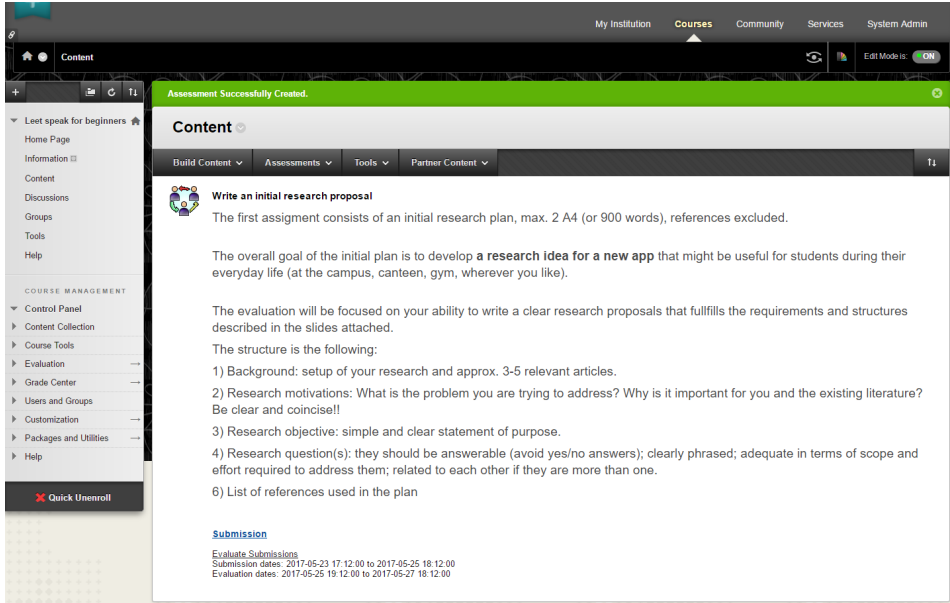


Figure A.3: The assessment "Write an initial research proposal" as shown inside the course content list

Submission

Figure A.4 shows the submission page, where students submits their initial assignment (the one who is going to be evaluated by other peers).

Upload assignment: Write an initial research proposal

ASSIGNMENT INFORMATION

Write an initial research proposal

The first assignment consists of an initial research plan, max. 2 A4 (or 900 words), references excluded.

The overall goal of the initial plan is to develop a **research idea for a new app** that might be useful for students during their everyday life (at the campus, canteen, gym, wherever you like).

The evaluation will be focused on your ability to write a clear research proposals that fullfills the requirements and structures described in the slides attached.

The structure is the following:

- 1) Background: setup of your research and approx. 3-5 relevant articles.
- 2) Research motivations: What is the problem you are trying to address? Why is it important for you and the existing literature? Be clear and concise!!
- 3) Research objective: simple and clear statement of purpose.
- 4) Research question(s): they should be answerable (avoid yes/no answers); clearly phrased; adequate in terms of scope and effort required to address them; related to each other if they are more than one.
- 6) List of references used in the plan

Assessment Files: Example_01.pdf

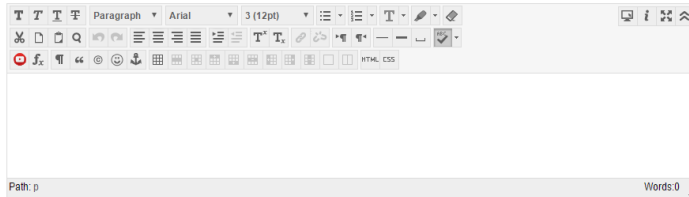
ASSIGNMENT FILES

Current File Name: No File

File to be imported:

ASSIGNMENT SUBMISSION TEXT

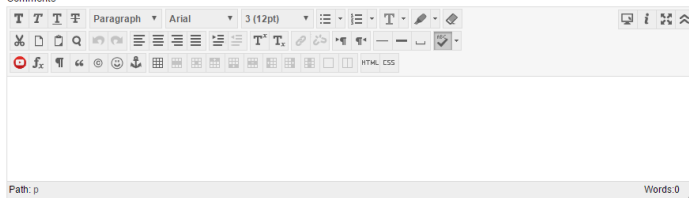
Submission text



A rich text editor interface for submitting text. It features a toolbar with various icons for text formatting (bold, italic, underline, strikethrough), paragraph alignment (left, center, right, justified), list creation (bulleted, numbered), indentation, and other editing tools. The font is set to Arial and the size to 12pt. Below the toolbar is a large empty text area. At the bottom left, it shows 'Path: p' and at the bottom right, 'Words: 0'.

ADD COMMENTS

Comments



A rich text editor interface for adding comments. It features a toolbar with various icons for text formatting (bold, italic, underline, strikethrough), paragraph alignment (left, center, right, justified), list creation (bulleted, numbered), indentation, and other editing tools. The font is set to Arial and the size to 12pt. Below the toolbar is a large empty text area. At the bottom left, it shows 'Path: p' and at the bottom right, 'Words: 0'.

Click **Submit** to finish. Click **Cancel** to quit without saving changes.

Figure A.4: The submission page

Students review page

After a student has submitted their submission and the submission time has passed, the submission is distributed to peer students for evaluation. Figure A.5 shows the student review page which contains all submissions the students should review.

Student Reviews

This are the submissions you have been assigned to review.

Name	Submission reviews	Status	Feedback on evaluation
Mark Hammington	Review Submission	Not reviewed yet	No feedback received yet

Displaying 1 to 1 of 1 items [Show All](#)

Figure A.5: The students review page, showing all submissions that the student should evaluate

▢ Evaluation by: Mark Hammington

Rubric Evaluation

Criteria	Scales		
	Points: 3	Points: 5	Points: 10
Structure	<input type="radio"/> Points: 3 - Structure points not clearly defined - One or more structure points missing	<input type="radio"/> Points: 5 - Good structure - All points covered	<input checked="" type="radio"/> Points: 10 - Clearly defined structure - All points covered
Requirements	<input type="radio"/> Points: 3 - Some requirements not met - Missing details (e.g. introduction missing its objective)	<input checked="" type="radio"/> Points: 5 - Points covers most important aspects - Weak text explanations	<input type="radio"/> Points: 10 - Good at covering all aspects - Has all requirements covered

Rubric score: 15

Verdict

Overall, the research proposal is interesting, however, the connection with related work is a bit unclear. Also I'd recommend avoiding strong statements in the introduction that are not grounded empirically and/or theoretically ("It is beyond reasonable doubt...").
Apart from this the assignment is well structured and contains all the necessary elements of a qualified answers. I have chosen to give a bit more points than the rubric, but not full score.

Final points: 18

Evaluation Feedback

Give a thorough feedback on the evaluation. In addition to a written feedback, specify a reflecting score.

For the toolbar, press ALT+F10 (PC) or ALT+FN+F10 (Mac).

Thanks for the feedback

I will certainly look into and change the strong statements in the text, and create a better connection with related work.

You feedback was informative, but maybe a bit short, I would have wished some more information on how well I had covered my explanation of the research behind efficient learning.

I gave you nearly full score, because it was a good feedback, but could have contained a more detail explanation.

Path: p

Words:75

Give Score: /5

[Submit](#)

Figure A.6: Part of the review of review page, which shows a student giving feedback on an evaluation

Appendix B

Unit and integration tests

Unit tests

	Description
ID	UNIT 2
Name	Group assignment
Date	April 25, 2017
Subject	create.jsp group display
Steps	In the create page under submission details: <ol style="list-style-type: none">1. Choose group submission.2. Confirm that all groups in the course is displayed in the list.3. Move the groups between the lists.4. Confirm that the correct groups are moved.
Results	Groups are displayed and moved correctly

	Description
ID	UNIT 3
Name	Rubric criteria
Date	May 02, 2017
Subject	create.jsp rubric
Steps	In the create page under submission instructions: <ol style="list-style-type: none"> 1. Click on add criteria. 2. Confirm that a new criteria is added in the rubric.
Results	Criteria correctly added to rubric.

	Description
ID	UNIT 4
Name	Add criteria scale
Date	May 02, 2017
Subject	create.jsp rubric
Steps	In the create page under submission instructions: <ol style="list-style-type: none"> 1. Click on add scale. 2. Confirm that a new scale is added to every criteria.
Results	Scale correctly added to rubric.

	Description
ID	UNIT 5
Name	Delete criteria scale
Date	May 02, 2017
Subject	create.jsp rubric
Steps	In the create page under submission instructions: <ol style="list-style-type: none"> 1. Click on delete scale. 2. Confirm that a the last scale is removed.
Results	Criteria correctly removed from rubric.

	Description
ID	UNIT 6
Name	Delegate grading
Date	May 03, 2017
Subject	create.jsp and instructor_review.jsp
Steps	In the create page under delegate grading: <ol style="list-style-type: none"> 1. Add a student to be able to grade the assessment submissions. 2. Confirm that the student can see the submissions in the instructor reviews page.
Results	Front-end completed - back-end not finished

	Description
ID	UNIT 7
Name	Upload assignment answer
Date	May 07, 2017
Subject	submission.jsp
Steps	In the submission page under assignment files: <ol style="list-style-type: none"> 1. Click on browse. 2. Choose a file to upload. 3. Confirm that the correct file is chosen. 4. Click on submit. 5. Confirm that the correct file has been uploaded.
Results	File selected and uploaded successfully.

	Description
ID	UNIT 8
Name	Choose rubric score
Date	May 08, 2017
Subject	assessment_review.jsp
Steps	In the assessment review page: <ol style="list-style-type: none"> 1. Choose a rubric cell in each row. 2. Confirm that the correct cell is chosen. 3. Confirm that the rubric score is updated correctly.
Results	Correct cells chosen and score updated successfully.

	Description
ID	UNIT 9
Name	ViewerJS PDF display
Date	May 08, 2017
Subject	assessment_review.jsp
Steps	In the assessment review page: <ol style="list-style-type: none"> 1. Confirm that correct the uploaded PDF from the submission is displayed in the ViewerJS.
Results	Correct PDF displayed.

	Description
ID	UNIT 10
Name	Other reviews
Date	May 10, 2017
Subject	assessment_review.jsp
Steps	In the assessment review page as an instructor: <ol style="list-style-type: none"> 1. Scroll to bottom of page. 2. Confirm that other reviews are listed (after adding reviews on the submission as students).
Results	Other reviews listed with correct values.

Description	
ID	UNIT 11
Name	Submit final score
Date	May 11, 2017
Subject	instructor_review.jsp
Steps	In the instructor review page: <ol style="list-style-type: none"> 1. Choose a submission. 2. Change the final score field. 3. Click on submit for that submission. 4. Confirm that the score is added in the Grading Center.
Results	Score added in Grading Center and status text turned green after submitting.

Description	
ID	UNIT 12
Name	Submission end date
Date	May 13, 2017
Subject	review_of_review.jsp
Steps	In the course content page: <ol style="list-style-type: none"> 1. Choose a submission after end date has passed. 2. Confirm that you are sent to the review of review page, and not the submission page.
Results	Correctly redirected to the review of review page.

Description	
ID	UNIT 13
Name	Student evaluations
Date	May 13, 2017
Subject	review_of_review.jsp
Steps	In the review of review page: <ol style="list-style-type: none"> 1. Confirm that all evaluations of the submission is displayed with correct information.
Results	Evaluations of submission correctly displayed.

Integration tests

	Description
ID Name Date Subject Steps	INTEGRATION 1 Create peer assessment May 04, 2017 create.jsp and course content In the create page: <ol style="list-style-type: none">1. Fill in all the different fields.2. Submit the assessment.3. Confirm that the assessment appears in the course content.
Results	Assessment successfully created and added in course content.

	Description
ID Name Date Subject Steps	INTEGRATION 2 Submit assessment answer May 07, 2017 submission.jsp In the submission page: <ol style="list-style-type: none">1. Upload submission file and fill in all the fields.2. Submit the answer.3. Confirm that the submission is created and stored.
Results	Submission successfully created and stored.

	Description
ID Name Date Subject Steps	INTEGRATION 3 Evaluation assignment May 07, 2017 create.jsp and student_review.jsp In the create page: <ol style="list-style-type: none"> 1. Choose a number of submissions to be evaluated. 2. Submit an answer to the assessment with as many students as possible. 3. Confirm that the submissions appear in the student_review.jsp list of submissions to evaluated.
Results	Submissions correctly assigned and added to list.

	Description
ID Name Date Subject Steps	INTEGRATION 4 Submit evaluation May 08, 2017 assessment_review.jsp In assessment review page: <ol style="list-style-type: none"> 1. Choose rubric scores, overwriting score and fill in some feedback. 2. Submit the evaluation. 3. Confirm that the evaluation is submitted.
Results	Evaluation successfully created and stored. Status text turned green to confirm the evaluation submission on the student review page.

	Description
ID	INTEGRATION 5
Name	Review of review
Date	May 10, 2017
Subject	review_of_review.jsp assessment_review.jsp
Steps	<p>In the review of review page:</p> <ol style="list-style-type: none"> 1. Fill in a feedback on an evaluation. 2. Click on submit. 3. Confirm that the feedback is created and stored correctly. 4. Confirm that the feedback is displayed for the evaluator in the assessment review page.
Results	Review of review correctly created and displayed for the evaluator.