Institute of Software Engineering
Software Quality and Architecture

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Master's Thesis

Interdisciplinary Composition of E-Learning Platforms based on Reusable Low-Code Adapters

Niklas Meißner

Course of Study: Softwaretechnik

Examiner: Prof. Dr.-Ing. Steffen Becker

Supervisor: Sandro Speth, M.Sc.
Dr. rer. nat. Uwe Breitenbücher

Commenced: April 1, 2022

Completed: September 30, 2022

Abstract

Electronic Learning (E-Learning) platforms or Learning Management Systems (LMSs) are becoming increasingly popular and, accordingly, are being used more and more by teachers at schools and university professors. They are used to digitally distribute educational material to students and provide the opportunity to, e.g., upload and collect assignments, solve tasks, and view grades. LMSs have been growing in popularity and are used alongside in-person lectures as an adjunct to self-study. Due to digital teaching during the COVID-19 pandemic, LMSs have increased in importance significantly. Even in after-pandemic times, with returning in-person lectures, it is hard to imagine teaching at universities without these platforms. The possibilities of working with the established LMSs are enormous. However, a closer look also reveals some negative aspects that were not considered in developing and using these platforms.

The existing LMSs lack individualization for lecturers of their courses and a motivating design for students. Plugins attempt to remedy this, but they are complex and time-consuming to use. Thus, the underlying problems are, on the one hand, that lecturers are limited in the design of their courses and, on the other hand, that students experience disadvantages in terms of motivation and interactivity.

This thesis aims to develop a concept for an e-learning platform that addresses these problems, supports lecturers in designing their courses, and motivates and assists students in learning. Under the aspect of generalization, a concept for a Software Product Line (SPL) was developed for the requirements of a wide variety of study programs, providing lecturers with a base platform and enabling them to use low-code adapters to design and modify their courses. In addition, the platform and a support team will assist lecturers in using the LMS and creating educational material.

For the conceptual design of the LMS, some already existing solutions and approaches were examined to address a similar problem. However, similar problems have been insufficiently solved or overlap with the problem statement of this thesis only to a limited extent. After a requirements analysis, the requirements were gathered and listed so that solutions could then be developed. The prototypical implementation of the concept '*Interactive Training Remote Education Experience (IT-REX)*' was used to design the base e-learning platform and to include gamification aspects. However, since IT-REX was designed for computer science and software engineering students in the first semesters, it had to be modified for a broader range of uses. To evaluate the approach of the concept, a case study was conducted in which a low-fidelity prototype of the concept was presented to lecturers and other experts in the field of higher education didactics, learning psychology, and vocational and technical pedagogy. Subsequently, a questionnaire was used to assess and evaluate the previously defined requirements.

The result of this elaboration is the concept for the e-learning platform with the corresponding prototype. Based on the feedback of the lecturers and experts, improvements and revisions could be identified. Furthermore, the evaluation helped to investigate how the platform's usability could be enhanced to improve the structuring and design of the courses by the lecturers. Finally, future developments and further investigations based on the concept were described.

Kurzfassung

Electronic Learning (E-Learning) Plattformen oder Learning Management Systems (LMSs) werden mehr und mehr zu einem festen Bestandteil der Lehre an Schulen und Universitäten und werden somit auch von Lehrern und Professoren zunehmend genutzt. Sie werden eingesetzt, um Lehrmaterial digital zu verteilen, aber auch um die Möglichkeit zu bieten z.B. Beiträge hochzuladen, Aufgaben zu lösen oder persönliche Noten einzusehen. LMSs werden immer populärer und parallel zu der Präsenzvorlesung als Zusatz für das Selbststudium verwendet. Durch die COVID-19 Pandemie bedingte digitale Lehre gewannen LMSs zunehmend an Bedeutung. Auch nach Rückkehr zu Präsenzvorlesungen sind die Platformen jetzt etabliert und nicht mehr aus der Lehre an Universitäten wegzudenken. Die Möglichkeiten mit den bereits existierenden und gängigen LMSs zu arbeiten, sind groß, aber bei näherer Betrachtung fallen einige Punkte negativ auf, die bei der Entwicklung und bei der Nutzung dieser Plattformen offensichtlich nicht berücksichtigt wurden.
Den bestehenden Plattformen fehlt es an Individualisierbarkeit, Anpassungsmöglichkeiten von Kursinhalten, sowie an einem motivierenden Design für Studierende. Plugins versuchen Abhilfe zu schaffen, sind aber sehr komplex und zeitaufwendig in der Anwendung. Die zugrundeliegenden Probleme bestehen somit darin, dass auf der einen Seite Dozenten in der Gestaltung ihrer Kurse auf der E-Learning Plattform eingeschränkt sind und auf der anderen Seite Studierende wenig zur Nutzung motiviert werden und sie in den Möglichkeiten der Interaktion stark eingeschränkt sind.
Ziel dieser Thesis ist es, ein Konzept für ein LMS zu entwickeln, das diese Probleme löst, die Dozenten bei der Gestaltung ihrer Kurse unterstützt und die Studierenden motiviert und beim Lernen unterstützt. Unter der Prämisse einer möglichst weitreichenden Einsatzmöglichkeit wird für die Anforderungen verschiedener Studiengänge ein Konzept für eine Software-Produktlinie entwickelt, bei der Dozenten eine E-Learning Plattform zur Verfügung gestellt bekommen. Mit Hilfe von Low-Code Adaptern können sie ihre Kurse nach eigenen Anforderungen und Ideen gestalten. Darüber hinaus soll es ihnen ermöglicht werden, durch die Plattform selbst und durch ein Unterstützungsteam die nötige Hilfe im Umgang mit der Plattform und der Erstellung von Lehrmaterial zu erhalten.
Zur Konzeptionierung des LMS wurden einige bereits existierende Lösungen und Ansätze untersucht, die versuchen, ein ähnliches Problem zu adressieren. Diese sind jedoch unzureichend gelöst oder überschneiden sich nur geringfügig mit der Problemstellung dieser Thesis. Nach erfolgter Anforderungsanalyse wurden die Anforderungen gesammelt und analysiert, so dass bedarfsgerechte Lösungen erarbeitet werden konnten. Für die E-Learning Plattform wurde sich für Elemente des prototypisch entwickelten Konzepts des '*Interactive Training Remote Education Experience (IT-REX)*' entschieden, da diese bereits wertvolle Gamification Aspekte umfasst und somit eine gute Grundlage bildet. Da IT-REX jedoch lediglich für die ersten Semester Informatik und Softwaretechnik konzipiert wurde, musste die Plattform für die Verwendung in anderen Studiengängen angepasst werden. Um die Ansätze des Konzeptes zu evaluieren, wurde eine Fallstudie durchgeführt, in denen Dozenten und Experten verschiedener Bereiche der Low-Fidelity Prototyp des Konzeptes vorgestellt wurde. Anschließend wurde mit Hilfe eines bereitgestellten Fragebogens das Konzept und die Umsetzung bewertet und evaluiert.
Das Ergebnis dieser Thesis ist das ausgearbeitete Konzept der E-Learning Plattform mit dem dazugehörigen Prototypen. Dabei brachten Rückmeldungen und Meinungen von Dozenten und Experten zahlreiche Vorschläge für Überarbeitungen und Optimierungen des Konzepts. Die Evaluierung trug zur Untersuchung bei, wie die Nutzbarkeit für Dozenten bei der Strukturierung und Gestaltung ihrer Kurse verbessert werden konnte. Abschließend wurden zukünftige Weiterentwicklungen und mögliche weiterführende Untersuchungen des Konzeptes beschrieben.

Acknowledgement

I would like to thank Steffen Becker, Sandro Speth, and Uwe Breitenbücher for their valuable feedback on this thesis and their general support in academic writing. Without your help, the concept of this thesis would undoubtedly have been described in a less detailed and thoughtful way. Our regular meetings were always an incentive and motivation to push the work forward. I would also like to thank Bernd Zinn and Charlotte Knorr, as well as Avni Qekaj, Katja Heere, and Maria Wirzberger, for their input and detailed feedback on the requirements and evaluation. Finally, I would like to thank everyone who gave me such extensive feedback on the concept and realization of my work to evaluate it.

Contents

1 Introduction **1**
1.1 Problem Statement 2
1.2 Solution Approach 4
1.3 Thesis Structure 6

2 Foundations and Related Work **7**
2.1 Literature Research Methodology 7
2.2 Foundations 7
2.2.1 Learning Management System (LMS) 8
2.2.2 Low-Code Development Platform (LCDP) 9
2.2.3 Software Product Line (SPL) 9
2.2.4 Gamification 10
2.3 Related Work 11
2.3.1 IT-REX - Vision for a Gamified E-Learning Platform 11
2.3.2 E-Learning Model for Teaching Mathematics 12
2.3.3 E-Learning Software Product Line 12

3 Conception of the E-Learning Platform with Low-Code Adapters **13**
3.1 Analysis and Requirements Engineering 13
3.2 Gathered Requirements 15
3.3 Overview of the Concept 20
3.4 Aspects of the Concept 22
3.4.1 Base E-Learning Platform 22
3.4.2 Low-Code Adapter 24
3.4.3 Interfaces and Data 26
3.4.4 Business Processes 29
3.4.5 Lecturer Support 32
3.5 Domain Model as UML 33

4 Implementation **35**
4.1 Overview of the Main Features and Architecture 35
4.2 Low-Fidelity Prototype Implementation 40
4.3 Used Tools and Technologies 48

5 Evaluation **51**
5.1 Overview of the Evaluation Process 51
5.2 Goal Question Metric (GQM) 52
5.3 Case Study 54
5.3.1 Study Design 54
5.3.2 Results 55

5.3.3 Discussion 61
5.4 Validation 62
5.5 Threats to Validity 63
5.5.1 Internal Validity 63
5.5.2 External Validity 64
5.5.3 Construct Validity 64

6 Conclusion and Future Work **65**
6.1 Results and Conclusion 65
6.2 Future Work 67

Bibliography **69**

List of Figures

1.1 Current use of e-learning platforms in digital teaching. 2
1.2 This thesis solution approach for the problem statement. 5

3.1 Requirements engineering process of this thesis. 14
3.2 Stakeholder overview. 15
3.3 Course structure in the e-learning platform as SPL. 20
3.4 Overview of the platform composition. 21
3.5 Layout and responsibilities of the base platform and different types of adapters. . 22
3.6 Categorization of a set of example adapters. 25
3.7 Concept of a suitable architecture with interfaces. 27
3.8 Dashboard data model as Unified Modeling Language (UML) diagram. 29
3.9 Business Process Model and Notation (BPMN) of course creation. 30
3.10 BPMN of adapter installation and linking. 31
3.11 BPMN of adapter adjustments. 32
3.12 E-Learning platform domain model. 34

4.1 High-level overview of the prototype architecture. 36
4.2 Interfaces between the frontend and the base platform microservices. 37
4.3 Interfaces between the frontend and the adapter microservices. 38
4.4 Overview of the low-code adapters and their mutual connections. 39
4.5 Implementation of course creation functions. 42
4.6 Implementation of the adapter overview functionality. 43
4.7 Implementation of the course overview functionality from lecturer's perspective. . 44
4.8 Implementation of course configuration functions. 45
4.9 Implementation of the adapter configuration functionality. 45
4.10 Implementation of the student course overview function. 46
4.11 Implementation of dashboard functionality for students. 47
4.12 Implementation of the course timeline function. 47

5.1 Evaluation process. 52
5.2 Goal Question Metric (GQM) for the evaluation. 53
5.3 Mindmap of the participants' feedback. 56
5.4 Questionnaire result for the appropriateness of the adapter approach. 58
5.5 Questionnaire result for the easiness of use. 59
5.6 Questionnaire result for the patience of the lecturers who set up their course. . . . 59
5.7 Questionnaire result for the process of using the platform. 60
5.8 Questionnaire result for improvement compared to established platforms. 60
5.9 Mindmap of the participants' feature requests. 61

Acronyms and Glossary

API Application Programming Interface. 27, 36, 37, 38

AR Augmented Reality. 67

BPMN Business Process Model and Notation. xi, 22, 29, 30, 31, 32

CSS Cascading Style Sheets. 49

E-Learning Electronic Learning. iii, v, xi, 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 25, 26, 28, 29, 30, 32, 33, 34, 35, 42, 51, 54, 55, 57, 58, 60, 61, 64, 65, 66, 67

GQM Goal Question Metric. xi, 51, 52, 53, 62, 65

HTML Hypertext Markup Language. 48

HTTP Hypertext Transfer Protocol. 37, 48

IT Information Technology. 4, 9, 14, 15, 18, 20, 32, 54, 58, 65, 66

IT-REX Interactive Training Remote Education Experience. iii, v, 1, 4, 11, 15, 17, 21, 64, 65

JPA Jakarta Persistence API. 48

LCDP Low-Code Development Platform. 7, 9, 20, 65, 67

LMS Learning Management System. iii, v, 1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 16, 17, 18, 19, 20, 22, 23, 24, 26, 28, 32, 34, 35, 51, 55, 58, 60, 62, 65, 66, 67

MVC Model View Controller. 37, 48

MVP Minimum Viable Product. 35

PoC Proof of Concept. 35, 52, 64

PSE Programmierung und Softwareentwicklung. 11

REST Representational State Transfer. 37, 38, 48

SPL Software Product Line. iii, xi, 9, 10, 11, 12, 20, 21, 66

SQL Structured Query Language. 37

UI User Interface. 4, 20, 23, 24, 28, 35, 37, 42, 48, 57, 58, 61, 64, 65, 67

UML Unified Modeling Language. xi, 6, 29, 33

VR Virtual Reality. 67

1 Introduction

Nowadays, it is common for lecturers at universities to use Electronic Learning (E-Learning) platforms or, in other words, Learning Management Systems (LMSs) for their digital teaching to complement their courses. For some time, they have been used to supplement face-to-face teaching to support students in their independent learning. However, since the COVID-19 pandemic, some lecturers have been forced to consider LMSs in their courses and use them in their digital teaching. Even today, after face-to-face lectures have largely returned, schools and universities are still trying to adapt and integrate digital elements into their lectures efficiently. However, these LMSs come with some drawbacks, including that students' motivation decreases due to social distance from other students [MBH+20].
Moreover, lecturers are massively limited in their options since teaching via digital media is more challenging than face-to-face. Not all teaching methods can be applied to e-learning platforms. Not only does the communication of educational material suffer, but also the scope and depth of what is taught compared to traditional face-to-face lectures [AA15, p.35-36].

'*Interactive Training Remote Education Experience (IT-REX)*', a project initiated by the University of Stuttgart, motivates students to support their studies by combining gamification and education. At the same time, it offers lecturers to structure their courses and educational material and provide them to students in an organized form. Quizzes and other gamification aspects offer students an additional incentive to acquire new knowledge while motivating them to repeat knowledge they have already learned. The e-learning platform IT-REX was designed to teach computer science and software engineering students in their first semesters [SBB+22].
Nevertheless, since the concept was developed exclusively for this use case and is not designed to be used domain-independently for many different study programs, its application in this thesis is limited. However, due to the features that IT-REX already brings, it was decided to design the concept with low-code adapters with the background of the IT-REX platform and to keep the aspects of gamification in consideration as well.

While structuring the educational material seems essential, the LMS must also be able to fully map the entire course content and be tailored to the lecturer's teaching. To effectively enable lecturers to do this and to seamlessly integrate all content in the LMS, it is important to create a concept for the composition of individual customizable components. To achieve this, lecturers' requirements must be addressed and supported in designing their courses within the LMS.

Finally, this thesis will investigate whether the concept of an e-learning platform with low-code adapters is applicable in higher education and whether such a platform will be easy to use for lecturers. In addition, experts in the areas of university didactics, vocational and technical pedagogy, and educational psychology will be interviewed in a case study for their views on whether such an e-learning platform adaptation is feasible.

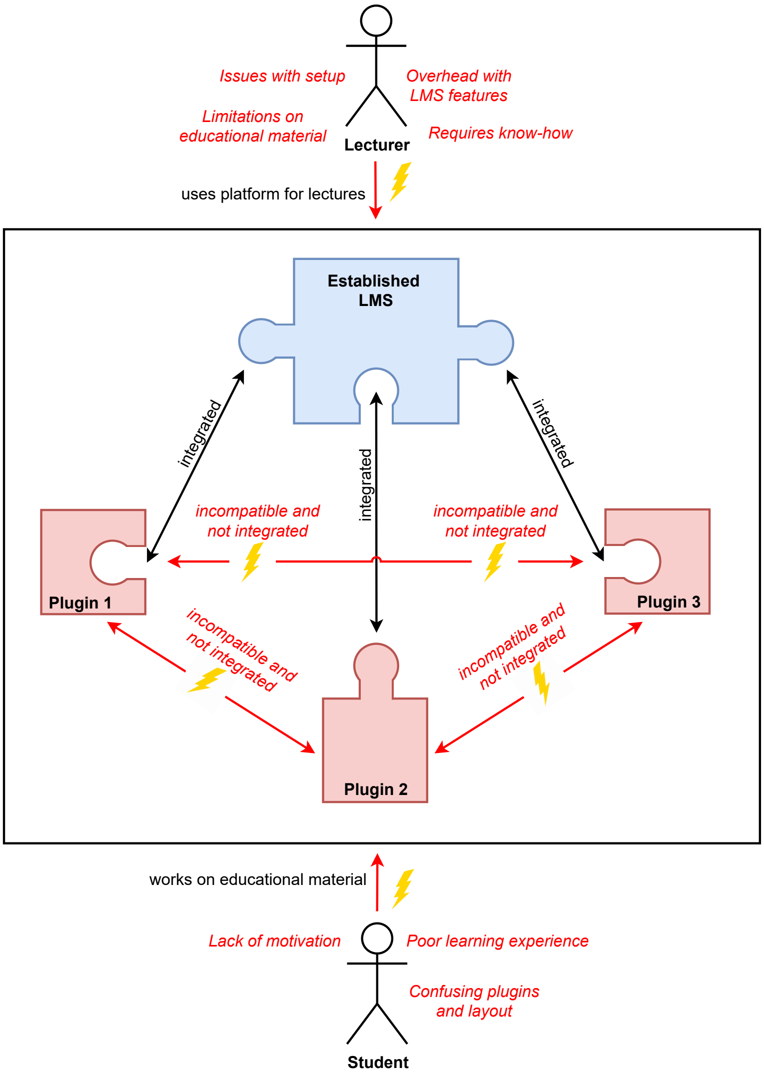

Figure 1.1: Current use of e-learning platforms in digital teaching.

1.1 Problem Statement

The overall problem is that universities' lecturers are limited in their options when using established LMSs. The content they can convey and teach to students in the face-to-face lecture cannot be presented entirely in the LMSs, or it is incomplete. Therefore, the established LMSs lack the functionalities and visualization options lecturers need for their face-to-face courses. Lecturers are thus forced to adapt their educational material to the corresponding environment instead of being able to adapt the e-learning platform to their needs and the material. For instance, lecturers from the field of linguistics should be able to use the e-learning platform as a tool for teaching languages and conducting group exercises. Lecturers should be able to integrate all required functionalities into the platform accordingly, e.g., flashcards for learning languages and quizzes with a combined meeting functionality for students to work on the group exercises. These functions are sufficient for the lecturers; thus, the course does not offer any other functionalities.

The streamlined design and the tailored content and functions to the requirements are lacking in established LMSs. As shown in Figure 1.1, these established platforms also offer plugins that can be installed on the platform. However, these plugins are standalones and are not integrated with other plugins, so they must be used as a distinct function. This disrupts the student learning experience since, for the example described earlier, lecturers would not be able to link and integrate the quiz function with a meeting function to work on group assignments. In addition, the plugins have been developed by third-parties and therefore require additional know-how from the lecturers.

There are two parts of the problem that are addressed in this thesis. *(1)* Established LMSs such as *Ilias* and *Moodle* try to remedy the stated problems with plugins and thus offer lecturers a more flexible platform. However, these plugins are poorly harmonized with each other and offer few opportunities for lecturers to adjust these plugins independently. *(2)* In addition, gamification is playing an increasingly important role in LMSs today to motivate students. Gamification must be consistent throughout the entire system if a benefit is to be achieved for students. Some plugins exist for established LMSs that try to integrate gamification into the system. However, these are isolated and do not connect to the platform itself, nor are they integrated with other plugins, so the benefits of gamification in digital teaching are limited [KAY14; PTP19].

Figure 1.1 shows the current use of established LMSs in digital education. On the one hand, some lecturers either use the platform to realize their entire course in it or use it as a supplement to in-person lectures. The overall problem statement describes lecturers as limited in the LMS. They often have to adapt their educational material to the platform's possibilities instead of the other way around. Possible plugins for designing individual courses are often complex, time-consuming, and difficult for lecturers to integrate independently. In addition, the customization and tailoring of these plugins to the requirements of the course is lacking, as these plugins do not provide any leeway for the adjustments of the basic functionalities and can only be modified as extended functionality and not in-depth, for example, by adapting configurations on a meta-level. On the other hand, students use the LMS to study with it. As outlined in the overall problem statement, digital teaching requires students to be more motivated, which is only partially encouraged by the established platforms. This is due to the social separation of students and the challenges of working independently with the platform, as well as increasing the motivation to work on educational material [MBH+20].
In addition, these platforms often cannot fully map the lecturer's material compared to the in-person lecture, so students end up at a disadvantage in terms of their knowledge, in contrast to students who have attended the lectures and exercises in person. For students in courses where lecturers only use the LMS as a supplement to their in-person lectures, this does not apply since the platform only provides additional self-study material.

In addition, Figure 1.1 shows how the established LMSs are structured and how they interact with plugins based on these platforms. LMSs such as *Ilias* and *Moodle* have a set of basic functionalities, which are shown in blue in the figure. Based on this, lecturers can add plugins to the platform to extend the basic functionalities. However, these lecturers often lack expertise in these plugins, making them difficult to use. The plugins can be, for example, calendar functionalities, meeting planners, or similar. These plugins, marked in red in the figure, can then be integrated into the LMS. The major problem with this is the heterogeneity among the plugins. Each plugin is fundamentally different and offers hardly any possibilities to interact with other plugins or to exchange data. There is no interaction among the plugins, so that no broader functionality can be achieved from a composition.

The problem statement leads to the following research questions of this thesis, which are validated in Section 5.4: '*Is the platform's high degree of customizability feasible for lecturers?*', '*Can a platform with low-code adapters be used in higher education under realistic conditions?*' and '*Is a platform with low-code adapters self-explanatory and easy to use?*'.

1.2 Solution Approach

This thesis presents the concept of the composition of e-learning platforms with reusable low-code adapters. In contrast to established LMSs, the lecturer should be given more options for customization and be able to tailor the platform precisely to their requirements. The composition of the base platform with low-code adapters brings a significant advantage over established LMSs for both lecturers and students. The low-code approach for integrating the adapters into the platform is helpful since lecturers in many fields of study can deploy and use the adapters without know-how. Especially for inexperienced lecturers in Information Technology (IT), the approach is precious. Low-code describes the creation of applications with little to no programming knowledge. Instead, lecturers can assemble functions independently with a rich User Interface (UI) to create a complete application that meets their needs. Also, deeper functionalities configurations do require little interaction with the source code; rather, customizations can be made within the graphical interface, allowing lecturers to independently customize their e-learning platform without expertise [Sal21]. How the low-code approach is used in this thesis is described in more detail in Section 2.2.2.

As with the previous use of LMSs, the base platform will offer basic functionalities that are domain-independent and can, therefore, be used by a wide range of study programs. Similar to the established platforms, a set of functionalities will be provided, enabling the lecturer to design the course without effort. These essential functionalities should not be limited to the functionalities of established LMSs. Generalizable functionalities are identified in the context of this thesis, which should be offered across domains. On the other hand, domain-specific adapters are offered for integration into the base platform. It should not only be possible to add platform-specific low-code adapters to the platform but also to use interfaces for other applications. An example of using external applications as an adapter is the *Automated Assessment Management System (ArTEMiS)* for the automated evaluation of source code in software development tasks [KS18].

Figure 1.2 shows the solution approach of this thesis. The base e-learning platform is not developed on the established platforms since they have some disadvantages, as described in the problem statement. Instead, the concept of this thesis is based on the prototypically developed platform IT-REX, as this already has some advantages for teaching [SBB+22].
The concept of IT-REX helps to analyze the domain-specific components for computer science and software engineering and to bring them to a domain-independent level so that they can be used in various non-IT courses. The main focus is to enable lecturers to adapt and tailor the components to their courses using low-code adapters. Furthermore, IT-REX also includes gamification features, which resulted from the requirements of lecturers gathered in a project by master's students at the University of Stuttgart. Thus, the base platform IT-REX already provides the foundation to support students' studies and motivate them with gamification. For these reasons, the concept developed in this thesis refers to some content from the platform IT-REX.

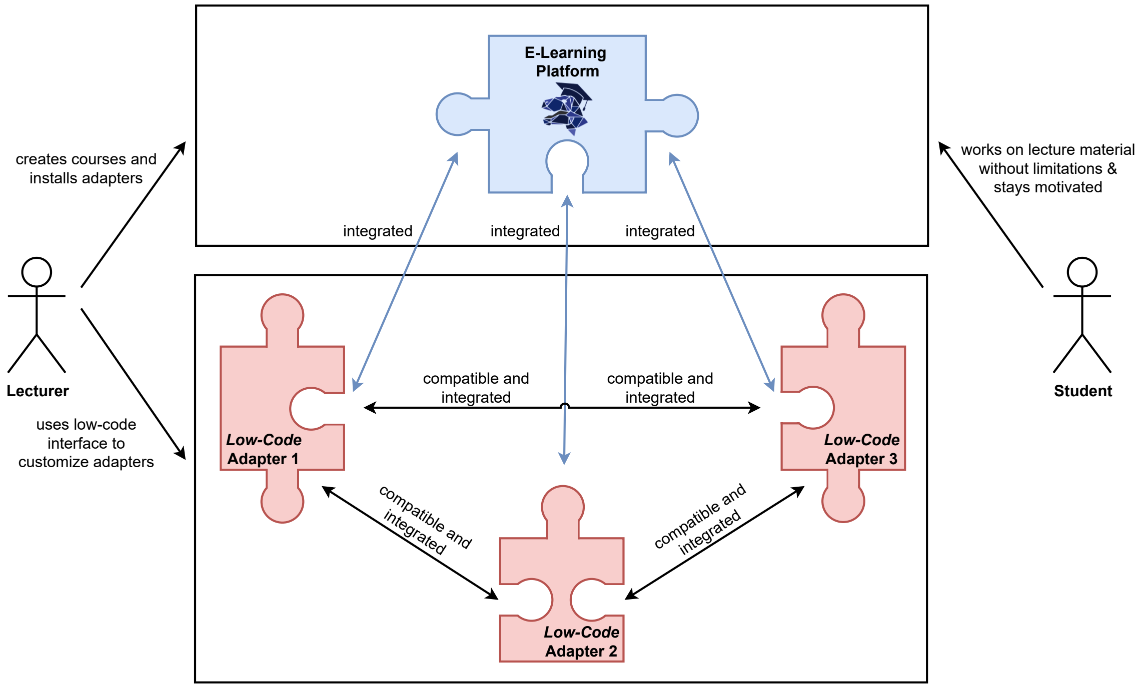

Figure 1.2: This thesis solution approach for the problem statement.

In addition, Figure 1.2 shows the structure and composition of the base e-learning platform and the adapters. The platform not only integrates the low-code adapters as such, but they are also compatible. They can be integrated so that the composition of various adapters provides new opportunities to the lecturers. All components should be as loosely coupled as possible and easy to configure so that data can be transferred from one adapter to another with minimal effort. The lecturer should be able to adjust any settings. The most important feature of this coupling and the data transfer between the adapters relates to the gamification that permeates the entire LMS. Gamification elements should not only be located on the top-level platform but also be included within the adapter functionalities.

To give lecturers the greatest possible flexibility in designing their course, the adapters should not only be easy to select and integrate into the base platform but should also be as customizable as possible. To this objective, there are three levels at which lecturers can customize the adapters. Firstly, integration of adapters as such, as they are offered in the established LMSs so that they can install them and use their functionality. On the other hand, there is the possibility of configuring and redesigning the adapters through configuration settings so that combined functionalities from multiple adapters can be precisely adjusted to the needs.

To evaluate and validate the concept of this thesis, we created a low-fidelity prototype. Based on this prototype, the concept's functionalities and implementation are evaluated by lecturers and experts with different technical and professional backgrounds. For this evaluation, a guided trial of the prototype was conducted, followed by a questionnaire. It emerged that lecturers would increasingly use the LMS and appreciate the customizability due to the possibilities of adapting and tailoring the platform to their courses. Based on this evaluation, the concept of the thesis is validated.

The contribution of this thesis includes several aspects:

- Derive components based on existing approaches, incorporating the requirements of lecturers and professors with didactic and pedagogical backgrounds.
- Creation of a concept taking into account the results of the research conducted and testing it in a real scenario.
- Evaluation of a low-fidelity prototype of the e-learning concept in terms of usability and applicability of the system by lecturers.

To summarize, a concept for an e-learning platform is created, which allows lecturers to tailor the platform to their courses and design it according to their requirements. For this purpose, the low-code adapters can be adjusted with configuration options to meet the needs of the lecturers.

1.3 Thesis Structure

The thesis is structured as follows:

Chapter 2 – Foundations and Related Work surveys the foundations and related work this thesis is based on.

Chapter 3 – Conception of the E-Learning Platform with Low-Code Adapters describes this thesis concept's requirements, aspects of the concept such as the base e-learning platform and the low-code adapter from both the business process and the interface and data perspectives, as well as the domain model as Unified Modeling Language (UML).

Section 4.2 – Low-Fidelity Prototype Implementation shows how the conceptual design of the e-learning platform with low-code adapters is put into a low-fidelity prototype to create a rough picture of the e-learning platform. This prototype can be used to evaluate the concept and further implementation.

Chapter 5 – Evaluation describes the evaluation of this work. First, the validation measure used is described. Then, this chapter explains how the metrics were collected, summarizes the results of interviewing various lecturers, and analyzes the outcomes. Finally, threats to the validity of this thesis are outlined.

Chapter 6 – Conclusion and Future Work concludes the results of this thesis and shows future research and implementation challenges.

2 Foundations and Related Work

This chapter explains the foundations and related work for this thesis. First, Section 2.1 describes the systematic procedure of the literature research. Then, in Section 2.2, some foundations are outlined, which are of great relevance to this elaboration. Finally, related work to the problem statement is described in Section 2.3.

2.1 Literature Research Methodology

For the literature research, the search engines *Google Scholar*, *Springer Link*, and *IEEE Xplore* were used due to the large number of literature databases behind them. The most important keywords for the research, which were also combined, were: '*E-Learning*', '*Learning Management System*', '*LMS in Higher Education*', '*Software Product Line*', '*Low-Code Development*', '*Low-Code Platform*', '*E-Learning Plugins*', '*Online Tutoring*', '*Digital Teaching*', '*Learning Motivation*', and '*Gamification*'. Forward snowballing was almost always used in the research for related literature. In most cases, the work cited in the literature reviewed was as exciting and related as the literature before it. The following points were applied as the main criteria for selecting appropriate literature:

(1) Is the literature related to the thesis?

(2) Can requirements of students and lecturers from different fields of study be derived from the literature?

(3) What kind of participants did the literature have and for what type of group was the main focus of the literature?

Criteria *(1)* and *(2)* both had to be met, while criterion *(3)* was optional to further investigate the literature. Once these criteria were resolved, the literature was added to the literature list. During the elaboration of this thesis, the literature list was continuously extended and improved.

2.2 Foundations

In this section, the foundations are described. First, Section 2.2.1 explains the foundations and main components of LMSs and shows the primary application areas. Section 2.2.2 presents the basics of Low-Code Development Platforms (LCDPs). Then, Section 2.2.3 describes software product lines, showing the variations of the different application domains in this thesis. Finally, Section 2.2.4 includes the basics of gamification, which is used to motivate students and increase learning success, and interest in the learning progress.

2.2.1 Learning Management System (LMS)

LMSs describe various applications, often very different, with many different models and purposes for these platforms. The definition is broad, so it must be clearly defined in which scope the term LMS is used in this thesis. These platforms can take different shapes and serve different purposes. Not only is e-learning used and advanced in schools and universities, but it is also used for commercial purposes or within companies for employee training. Thus, e-learning defines purely digital learning on a computer or other device, regardless of what content is taught or what group of users can access the material [BBZ04].

As different as LMSs can be, the fundamental functionalities, especially in the context of schools and universities, are similar for all platforms. Typically, they include access to educational material and the ability to check the gained knowledge with tests and exercises. In addition, they provide the functionality to communicate and collaborate between students. For lecturers, the platform typically offers the possibility of course management and assessment facilities. The most common LMSs in higher education are the open-source platforms *Ilias* and *Moodle*, as well as the commercial platform *Blackboard* [Pio10].

Most LMSs are web applications that can be accessed in a web browser. Each learner requires a stable internet connection and respective devices to access the platform. LMSs comprise three essential criteria that determine the acceptance and usability of the platform [BBZ04]:

(1) Effective learning of the student

(2) Effective creation of educational material by teachers

(3) Economic use of educational material in organizations

(1) The effectiveness of student learning is essential, as this is the primary goal of using a LMS. Therefore, the platform must provide clear guidelines on what educational material is available and how it can be accessed. The platform is not responsible for the content, as this is created by teachers or, in other cases by the providers. However, the presentation and the ease of use also contribute to the students' effective learning. *(2)* On the other hand, effective learning of the students requires suitable educational material from the teacher. For this, a LMS enables the teacher an effective and easy creation of content. This includes not only offering the various functions for uploading and creating content but also the simplicity and usability of the platform. Only when these aspects harmonize with each other, an effective use of the platform is possible. *(3)* The economical use of the content in organizations is as important as the creation itself. The content must be applicable and serve its purpose, that the educational material in the LMS is beneficial.

The previously mentioned open-source LMSs in higher education allow plugins to be added to supplement the functionalities offered by the platform. In the context of this elaboration, such a feature is used to extend the platform with adapters. With these, the platform can meet the needs of the lecturers.

2.2.2 Low-Code Development Platform (LCDP)

Low-Code Development Platforms (LCDPs) enable the development and deployment of fully functional software applications using advanced graphical user interfaces, and visual abstractions that require minimal or no procedural code [Was19]. By using low-code platforms, people with no programming experience can build their software applications without the help of multiple developers who usually used to be involved in the entire development of fully functional applications. Furthermore, this allows to focus on the application's business logic being specified instead of dealing with unnecessary details related to setting up the required aspects [BRH20; SIDP20; WFZS21].

The use of LCDPs enable the development and deployment of applications and functionalities in a short time without having to use source code. Adjustments cannot only be implemented in a short time but also less IT staff is required after low-code elements have been developed in a way that they can be understood by users from any application area. The low-code IT team that develops the low-code features works in the background, implementing functions and releasing them to a wide range of users. In traditional development teams, functions would be explicitly developed. However, they would not be deployable across the entire landscape, nor would end users be able to customize the functionalities on their own [PBSG19].

The global leading low-code platform providers include *Microsoft PowerApps*, *Google App Maker*, *Salesforce*, *TrackVia*, and *Appian* [Was19]. These platforms enable the rapid implementation and conversion of business processes into low-code so that functions can be developed with the help of a graphical user interface that, as a whole, makes up a standalone application. Neither the implementation of code nor a deeper understanding of the code behind it is required.

In the context of this thesis, lecturers should be enabled to assemble an application with the functionalities they require independently. For this purpose, they will be able to compose functionalities with the help of the graphical user interface and thus individualize their courses. Furthermore, by configuring parameters, lecturers can customize the individual functionalities. In addition, low-code interfaces should be able to ensure the customizability of the functionalities by being able to manipulate or extend the source code.

2.2.3 Software Product Line (SPL)

A Software Product Line (SPL) is a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission developed from a standard set of core assets in a prescribed way [Don14].
For the administration of a SPL, models represent all possible products that can be derived from it. The most common models used for this purpose are feature models. Feature models graphically represent the common and variable features in a SPL [RGGB19].

SPLs are also increasingly used in the field of LMSs, as they solve the problem of bad reconfiguration and variability. E-Learning platforms can, therefore, be developed with a wide variety of settings and thus adapt to universities on the one hand and students on the other. In addition, using SPLs in the area of LMSs increases the reusability of existing functionalities [AGB17]. LMS development with SPL is also beneficial for lecturers, as they can familiarize themselves with standardized business processes and a uniform design.

However, SPLs not only have advantages in the area of e-learning but also have obstacles that need to be dealt with separately. Problems can arise, such as dealing with non-technical stakeholders, the overhead of process diversity and version management, and domain-specific challenges [CN13].

To classify the use of a SPL in the context of this thesis, the diversity of LMSs must be considered. The required applicability of the platform in different fields of study makes using a SPL reasonable. These demands require different features, which can be covered by a SPL. Nevertheless, the extent to which SPLs can be applied to these specific features must be considered since courses of all kinds should remain within one LMS, and the SPL should thus be scoped within the platform.

2.2.4 Gamification

Gamification is an integration of game elements and game thinking into activities that are not games. Nowadays, students have grown up with digital technologies and have different learning styles, a new attitude towards the learning process, and higher demands on teaching and learning. Lecturers face new challenges and have to solve critical issues related to adapting the learning process to students' needs, preferences, and requirements. Lecturers must use different teaching methods that enable students to participate in their learning with great motivation, and commitment [Coo13; KAY14]. Gamification also implies social play and interaction with other people. When they perceive social presence, people naturally react socially and feel emotions such as empathy or anger [Mun11].

Gamification is still considered an emerging technology, which has still not fully exploited its potential [San15]. In higher education, gamification in e-learning platforms has been shown to improve students' outcomes and help them set goals for their learning [BHH20]. This is due to, among other things, the change in the way students learn through gamification. Gamification expands learning methods, putting students in new situations where they receive continuous and instructive feedback. These benefits lead to a continuous evolvement of LMSs in this direction of gamification [SMYK14].

Another important aspect of gamification is motivation. On the one hand, there is intrinsic motivation, which the student brings up independently to learn something, and on the other hand, extrinsic motivation, which is driven by rewards or sanctions. The interest generated by a task plays a decisive role in the motivational orientation of the student. The more attractive the task, the more interest, and curiosity arouses in the student [Dev97]. While the increased intrinsic motivation of a student is not necessarily associated with a positive outcome [FKY18], the motivation would drive the student to learn and thus increase the chances of improvement in the future [BN20].

In the context of this thesis, it is mandatory to focus on gamification features to provide and develop a modern platform. Furthermore, gamification has become common and is increasingly used in higher education, so the concept developed in this elaboration aims to motivate students to learn. Tasks should be designed so that their intrinsic motivation is encouraged to work on topics and educational material independently.

2.3 Related Work

The following section deals with modern e-learning platforms and the attempt to gamify them on the one hand and integrate them into a SPL on the other so that they can serve a wide range of possible application areas. First, Section 2.3.1 deals with the IT-REX platform and its original conception for computer science and software engineering students in their first semester to learn a programming language. Section 2.3.2 refers to an e-learning model that was applied and used in mathematics. Finally, Section 2.3.3 deals with an already implemented platform in the context of a SPL.

2.3.1 IT-REX - Vision for a Gamified E-Learning Platform

The IT-REX concept was created as a vision of an e-learning platform that would revolutionize digital teaching in the field of computer science. Due to the COVID-19 pandemic, a solution had to be found to keep students who were forced to study at home motivated despite the absence of lectures. The focus was on a LMS that addresses the higher education of computer science courses in the first semesters. A demand was identified since the first semesters are essential for computer science students as they build the foundation for the following courses. The goal was to equip the LMS with functionalities that, on the one hand, motivate students and reward them for learning with gamification elements and, on the other hand, extend the possibilities of the currently used platforms *Ilias* and *Moodle*, e.g., with new quiz types and a learning structure [SBB+22].

The vision also considered why the currently used LMSs are not being expanded with the existing plugins. The plugins for structuring a course as well as for checking source codes, are not simple to integrate and often lack an easy-to-use interface. As a consequence, lecturers mostly do not use them [SBB+22].

IT-REX was prototypically developed as part of a software project by master's students of the University of Stuttgart in software engineering. The platform is for both students and lecturers. The prototype of IT-REX is designed to support students in their learning, motivate them to continue learning, and show them how they are doing in their courses. For lecturers, the platform provides a solution for scheduling and uploading content such as lecture videos, scripts, and other types of documents related to their courses. Unfortunately, the platform IT-REX is primarily targeted at computer science and software engineering, specifically the '*Programmierung und Softwareentwicklung (PSE)*' course, which limits its applicability [Uni12; Uni20a; Uni20b].
The 'PSE' course at the University of Stuttgart teaches programming to students in their first semesters. During COVID-19, it was challenging to digitalize these lectures and exercises, which normally take place in class, and to offer them virtually in an e-learning platform [SKRB22]. Due to the problems that arose, the platform IT-REX was subsequently pushed forward.

The most crucial role of IT-REX for this thesis is to serve as a foundation to analyze the components related to computer science and software engineering and to generalize them for the SPL of the LMS so that they can be deployed and used across domains. The already developed prototype can be used as an example for possible implementations. This elaboration is partially based on the demands that have already been identified in the concept of the IT-REX.

2.3.2 E-Learning Model for Teaching Mathematics

The paper by Jeong Yong Ahn and Akugizibwe Edwin on *'An e-learning Model for Teaching Mathematics on an Open Source Learning Platform'* [AE18] develops and presents an e-learning model for teaching mathematics. Thereby, the fundamental aspects of digital education are examined, and learning theories such as social constructivism, social realism, and social connectivity are combined with the subject of mathematics. The model designed by the authors can enhance LMSs for math tasks and make the subject more exciting to learn [AE18].

During the model creation, the authors focus on the three pillars of teaching. These pillars are motivation, context, and interaction. For this, the e-learning model gives more responsibility to the lecturers to create more innovative educational material which fits the student's abilities [AE18].

For this thesis, the paper is of great interest, as it surveys similar issues in current e-learning platforms and aims to find a solution approach for the use in mathematics. However, the paper's scope is primarily limited to mathematics and thus cannot be used domain-independently. Nevertheless, the requirements for a modern e-learning platform provide a reasonable basis and synergies with the demands placed on e-learning platforms in this elaboration.

2.3.3 E-Learning Software Product Line

In the case study by Pablo Sanchez et al. on *'Software Product Line Engineering for e-learning Applications'* [SGZ12] the authors use the LMS Moodle and its 786 existing extensions and plugins at that time to examine how the market of extensions can be put into a software product line. Due to a large number of plugins, many systems are similar but not uniform. Therefore, the authors believe that using a SPL engineering approach to develop auxiliary applications for e-learning platforms can be of great benefit and illustrate how to create a SPL for a family of applications whose goal is to extract knowledge from log data stored in the LMS [SGZ12].

Pablo Sanchez et al. focus on the data mining level with the '*E-Learning Web Miner*', which is supposed to provide lecturers with information that the platform accumulates. This is implemented with the help of a SPL. Thus, adaptations to a product are carried out by the software itself, and changes can be implemented without additional interaction. To evaluate the idea, the authors implemented various adjustments to the application automatically with the help of the SPL [SGZ12].

The paper's requirements for using LMSs are similar to the requirements for the platform in this thesis, but this approach is based on the Moodle platform with already existing extensions and plugins. Moreover, the authors focus on the application of the E-Learning Web Miner and do not consider any domain-independent features that students can use in the end. Nevertheless, the general approach of SPLs in the e-learning environment is of utmost interest for this study.

3 Conception of the E-Learning Platform with Low-Code Adapters

This chapter describes the solution approach of this thesis, a concept for an e-learning platform with low-code adapters that solves the problem described in Section 1.1. First, a requirements engineering process was conducted to gather requirements, which is explained in Section 3.1. Then, these gathered requirements are written down in Section 3.2 and include user requirements from the lecturer and student perspectives, as well as general requirements for the LMS in the areas of the base platform, gamification, and extensibility. In addition, some non-functional requirements were elaborated, which the LMS has to fulfill. Based on the gathered requirements, a conceptual design was created to solve the problem. Next, Section 3.3 briefly summarizes this concept in an overview. Subsequently, Section 3.4 outlines the aspects of the concept and describes the e-learning platform and the low-code adapters. Among other things, the business processes of these adapters are explained, and the technical integration into the LMS is described from the perspective of interfaces and required data. Finally, Section 3.5 provides an overview of the domain model of the concept.

3.1 Analysis and Requirements Engineering

To analyze the requirements of the objectives of this thesis, a requirements engineering process was conducted. The goal of this requirements engineering was to determine quality requirements for the system that need to be implemented in the platform. These requirements must be clear, consistent, modifiable, and traceable. This requirements engineering process is one of the most important tools for gathering the requirements of software [PSR10].
The requirements engineering process is shown in Figure 3.1. The process consists of four sections, from which the conceptual design of the LMS is derived at the end, or rather the requirements were elaborated in a way that they could be used for the conceptual development. In general, this was an iterative rather than a completed process, as additional requirements were derived from each validation of requirements, thus starting the process over again.

Requirements Elicitation: The first step was the requirements elicitation, in which initial requirements were gathered through brainstorming and needs assessments. In addition, identifying stakeholders was also part of the first step. From the requirements elicitation, it was found that several key stakeholders played a major role in this thesis. An overview of the identified stakeholders is shown in Figure 3.2. One key stakeholder is the lecturer, who has to set up and configure the platform and upload content. On the other hand, the student who uses the LMS in the end. In addition, the platform must be considered from a didactic and pedagogical perspective as well as from the perspective of educational psychology. For this purpose, experts from these fields were also involved.

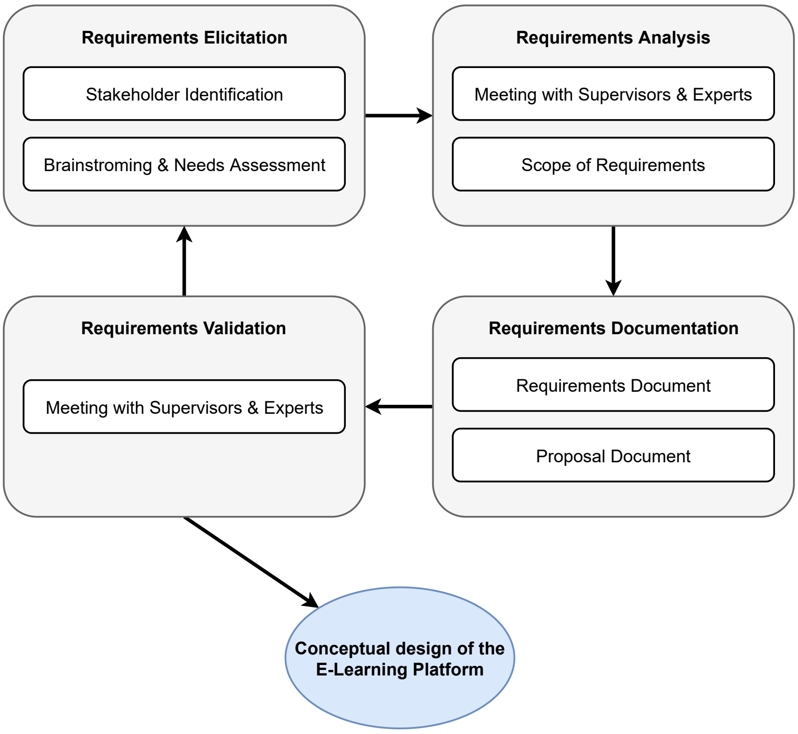

Figure 3.1: Requirements engineering process of this thesis.

This thesis focuses mainly on the stakeholder of the lecturer since their work should be made as easy as possible, and the platform should support their lectures. In addition, results from this section revealed that there are two classes of lecturers for the requirements. The lecturers who are not related to IT at all and, therefore, depend on any support for the configuration, and the lecturers who are IT related and, therefore, can extend and individualize their platform on their own.

Requirements Analysis: In the requirements analysis phase, the previously gathered requirements were examined to determine whether they fit into the scope of the concept of the e-learning platform and whether they are suitable for the context of this thesis. This selection reduced the already identified requirements to the most important ones. In addition, requirements, as well as the focus on the core content, were agreed upon in joint meetings with the supervisors of this thesis to set the frame for the conceptual elaboration. Finally, the requirements were reduced to the aspects of a base platform and expandability by using adapters so that they could be specified precisely.

Requirements Documentation: In the third phase, the requirements documentation and the previously gathered requirements for the e-learning platform were listed. On the one hand, these were recorded in a requirements document, which was continuously adapted and extended. On the other hand, the requirements for the system were already part of the proposal document. This proposal was created before this thesis to describe initial approaches and define its scope. Combining both, a list of requirements was created, which was adapted various times in the requirements engineering process. Many of the requirements were formulated in the form of user stories from the perspective of lecturers and students. For their documentation, *Microsoft Excel* was used, in which the demands were created and updated through the requirements engineering process.

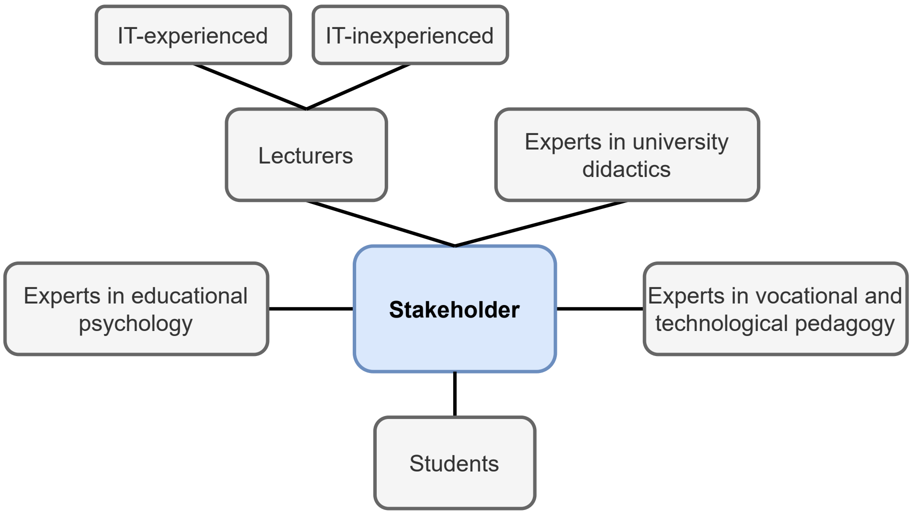

Figure 3.2: Stakeholder overview.

Requirements Validation: The validation of the requirements was the fourth and final phase of the requirements engineering process. In this phase, the demands were cross-checked by experts from the field of pedagogy and didactics to ensure that they were appropriate. For the input of professors and other lecturers, the needs of the linguist Katja Heere as a IT-inexperienced lecturer were included. These needs were already included in joint discussions during identifying requirements for the platform IT-REX. The requirements of the supervisors of this thesis, Professor Steffen Becker, Sandro Speth, and Dr. Uwe Breitenreiter, were represented as IT and software-related lecturers. On the other hand, the professor for vocational education, Bernd Zinn, and his Ph.D. student, Charlotte Knorr, were interviewed. Their demands were gathered from their perspectives as well. Both have IT experience due to their working area IT pedagogy. Therefore, the collected requirements were validated and discussed with them from a pedagogical point of view. From these discussions and the insights gained, additional needs emerged that were further integrated into the requirements engineering process and thus iterated. Ultimately, the conceptual design of the e-learning platform was started from the demands gathered. The requirements obtained from the process are described in Section 3.2.

Figure 3.2 provides an overview of the stakeholders. Among the stakeholders mentioned above, the requirements were also considered by the university didactician Avni Qekaj and the education psychologist Maria Wirzberger. To complete the list of stakeholders, the requirements of the students, which are relevant but not the main focus of this thesis, were also considered.

3.2 Gathered Requirements

As a result of the requirements engineering process, several requirements were gathered. This section highlights the gathered requirements, most of which were written in the form of user stories, but also included many general requirements for the base e-learning platform, gamification, and extensibility. The user stories are divided into two main sections: a lecturer's and a student's view.

Within these section, the user stories address requirements for the platform, as well as customizability and adaptability features. In the following, all functional requirements for creating and using the LMS are described. In addition to that, some non-functional requirements are listed as well.

General Platform Requirements

The following describes general requirements for the e-learning platform. The focus is on the requirements for the base platform, gamification, and extensibility. These requirements were established with a pedagogical background, which the platform has to fulfill to be beneficial for the lecturers and students.

Base E-Learning Platform

- Established LMSs already cover the fundamental needs of lecturers and students, so the base platform should positively distinguish through gamification.
- The e-learning platform must ensure heterogeneity. That means weak and good students must be treated differently to balance the unequal knowledge.
- The e-learning platform must offer both the cognitivist phase, in which the students are provided with information and have to process it, and the constructivist phase, in which the students find the solution on their own.
- The base e-learning platform needs a large set of standard features to integrate gamification in the best possible way and make it coherent for an entire course.
- The user experience must encourage and motivate studying.

Gamification

- The e-learning platform must encourage activities according to the student's knowledge and not reward them for guessing or random selection.
- At the beginning of a course, the students' competencies must be measured by the platform. Then gamification has to be used for students' weaknesses (adaptive support).
- Gamification for student collaboration and group work must be given a clear frame and boundaries by lecturers.
- The e-learning platform must include the three pillars of gamification. These are mechanics, aesthetics, and storytelling.
- The e-learning platform must allow domain-specific 'storytelling' gamification components, which must be integrated into the platform.
- Gamification elements must be able to be integrated and modified in low-code adapters.

Extensibility

- Low-code adapters must be able to be added to the base e-learning platform, which extend a course in a domain-specific way.
- Low-code adapters must be different from plugins for established LMSs in terms of customizability and gamification.
- External native applications must be able to be integrated or be offered an interface.
- Low-code adapters must be given a limited range of activities within the e-learning platform, as the advantages of gamification disappear and become inconsistent with too much modularity.
- Low-code adapters must be simple to add by the lecturer to save time.

Lecturer User Stories

The following describes the lecturer's user stories. First, user stories with requirements for the e-learning platform, and then the needs for extensibility and customizability are covered. Some of the requirements are based on the features of the IT-REX platform, which in any case need to be integrated into a standardized platform.

E-Learning Platform

As a lecturer, I want to ...

- use the essential functions of LMSs, e.g., uploading educational material.
- upload my educational material on the e-learning platform, as I am used to the established LMSs, even if I do not want to use the low-code adapter feature.
- be able to plan my course content and assign weeks as soon as the course is created (IT-REX functionality).
- be able to structure my course and allow students to unlock new educational material through learning successes (IT-REX functionality).
- be able to use elements of gamification to motivate my students as soon as the course is created (IT-REX functionality).
- carry out interactive group work in the e-learning platform and provide a framework for this.

Extensions with Adapters

As a lecturer, I want to ...

- extend the basic functionalities of the e-learning platform with adapters, as with plugin in established LMSs.
- tailor my course in the e-learning platform to my lecture and the practice session.
- be shown recommended adapter bundles for installation already during course creation after specifying the study subject.

- be able to manually select during course creation which adapters I want to install directly that are required for my course.
- get an overview of which adapters are integrated in my course.
- be able to remove installed adapters from my course environment at any time.
- be able to add more adapters to my course at any time.
- integrate third-party applications into the e-learning platform so they can be included in the platform.
- integrate third-party applications (i.e. *ArTEMiS* [KS18]) into the e-learning platform, so that they can be used by the platform.
- get support when I do not have IT skills or time to do so and need help customizing adapters.

Student User Stories

In the following, the user stories of the students are described. It is presumed that the students can use the platform like established LMSs, and thus the standard features of these platforms do not need to be mentioned further.

As a student, I want to ...

- use an easy-to-use e-learning platform and take advantage of the features of established LMSs.
- avoid having to use a different platform and tool for each of my courses.
- receive clear instructions from lecturers on how to use the e-learning platform for a course.
- receive the same education via the e-learning platform that I would typically receive in-person.
- use the e-learning platform as a supplement to in-person lectures to track my learning progress.
- be able to work on the assignments in the e-learning platform in the same way as the on-site exercises.
- be able to collaborate with other students in the e-learning platform as if we were working on a task on-site.
- experience competence, so that the tasks in the e-learning platform do not overwhelm me, and I can see how I develop through my knowledge.
- experience social inclusion so that I can learn and educate myself through group work and collaborating with other students.
- experience autonomy, so I feel that the knowledge I have learned allows me to create something independently.

Non-Functional Requirements

There are also some non-functional product quality requirements based on the ISO25010 quality model [IEC11], which are described below:

Usability
The system should be easy to operate, control, and use appropriately. The LMS must be easy to use and self-explanatory (operability). Lecturers need to be shown clearly which extensions they can add to the system and where they can make adjustments. It must also be easy to make modifications to the course structure, especially in the area of gamification. Students must always know where they are in the system and where they can navigate further. Mainly when different courses use different adapters, this can confuse them. It must also be immediately apparent to lecturers and students what functionalities the system offers.

Portability
The system should be effectively and efficiently adaptable for different courses (adaptability) and quickly set up. The LMS should be able to be used by courses from different fields. It should offer a large set of standardized functions and the possibility to tailor and individualize the e-learning platform for a specific course.

Maintainability
The system should be modular (modularity) and modifiable to change components with minimal impact on other components without introducing defects or degrading existing product quality. These characteristics start with the concept. All components of the e-learning platform should be individually modifiable (modifiability) and extensible.

Reliability
The system should be operational and accessible when needed for use (availability) and recoverable in case of failure (recoverability). The LMS must always provide correct and reliable results. If the system produces erroneous output, the e-learning platform should preferably not produce any output until the system is recovered.

Performance Efficiency
The system should have appropriate response and processing times when performing its functions. Adjustments to the system by lecturers should happen in near real-time so that a good user experience is maintained. Lecturers should be able to see adjustments directly in the e-learning platform to make interaction with the system as pleasant as possible. Otherwise, the system would lose acceptance and prestige.

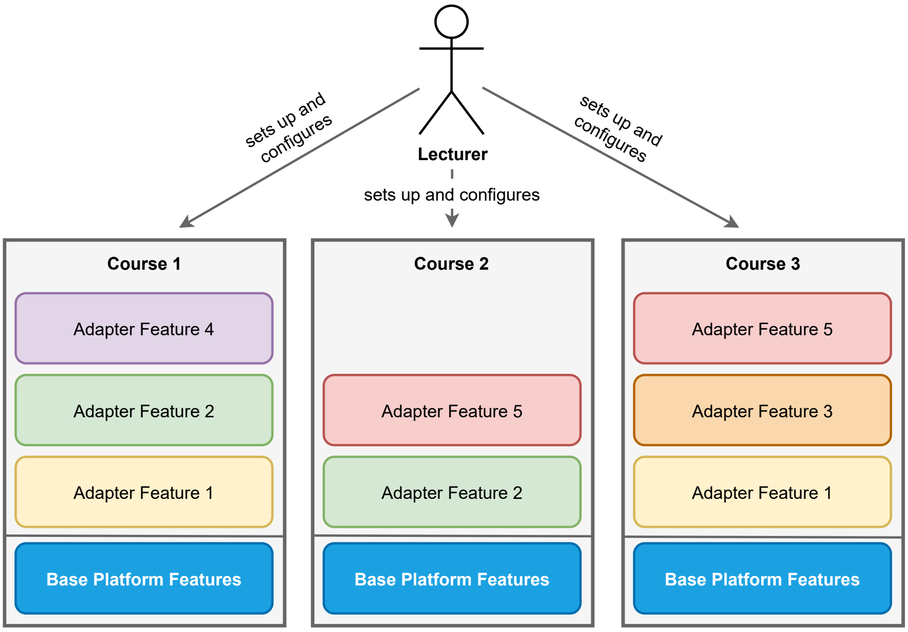

Figure 3.3: Course structure in the e-learning platform as SPL.

3.3 Overview of the Concept

The concept deals with a solution idea for e-learning platforms and how they can be tailored to the needs of lecturers' courses. Based on the requirement engineering process and the previously described requirements, a need for a highly customizable LMS emerges. For this purpose, the concept proposes to design a SPL for individual courses. The structure of this SPL with the deployment of functionalities within a course is shown in Figure 3.3. A base platform is provided for the lecturers, which offers domain-independent basic functionalities. All additional domain-specific functionalities and demands are covered by adapters that can be installed into the platform. This turns the platform into a SPL since a completely different platform is provided depending on the course type. It essentially belongs to the same product family.

Since lecturers are expected to integrate the adapters into the platform independently with the help of an intuitive UI, the e-learning platform is designed according to the LCDP. Therefore, even non-IT lecturers, who do not know the IT processes and developments behind these adapters, should still have the opportunity to design and extend the platform. Thus, the low-code approach was chosen for the integration of these adapters.

However, lecturers must be facilitated in uploading, structuring, and providing educational material to students, and students should feel equally comfortable and motivated to learn on the platform. To stand out from established LMSs, a high degree of customizability and the integration of gamification elements are of great relevance. The gamification aspect was elaborated within the concept from discussions with professors and experts from vocational and technology pedagogy. To implement gamification elements in a customizable platform in a meaningful way, these elements must be used heterogeneously throughout the whole platform. This means that if the lecturer decides to use gamification in the course, then the gamification must influence or be linked to all other adapters throughout the course to make gamification meaningful and to be able to use its benefits.

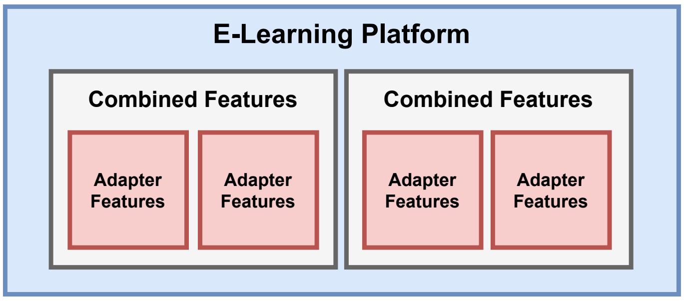

Figure 3.4: Overview of the platform composition.

Figure 3.4 shows an overview of the composition of the e-learning platform. The installable adapters are the smallest unit of the platform and contain basic functionalities. These are, for instance, functionalities for video uploads or the creation of exercises for students. The adapters can also integrate information and functions from third-parties, and other native applications, which provide an interface for the e-learning platform. Compatible adapters can unlock additional combined functionalities when installed in the platform. These combined functionalities can then be used in the platform. For instance, this can be a chat functionality when watching lectures if a feasible chat adapter and a video adapter have been installed in a course together. Thereby, more than two adapters can be connected and enable combined functionalities. The e-learning platform is thus built individually depending on the course and the adapters installed. Each course is, therefore, determined by the adapters installed and can be used according to the needs of the lecturers. Each course thus has very individual functionalities tailored to a course in addition to the domain-independent functions. Thus, the e-learning platform is considered as an SPL.

Since not only lecturers benefit from the concept developed in this thesis but also the students, some elements are included in the concept for the improvement of the platform from the students' perspectives. Therefore, the e-learning platform is designed to provide students with a dashboard as the home page of a course. This dashboard consists of individual widgets filled with information and functionalities by the individual adapters. Through the quick information on the home page, it should be possible for the students to get direct and personal information such as learning progress, following tasks, or assignments displayed. In this way, students know directly on a course's home page, e.g., what the subsequent assignments are or what the learning progress is. This content is displayed by the adapters installed in a course, can be customized as needed, and is thus determined by the lecturer's settings and configurations.

For the integration of gamification into the e-learning platform, the content of the project IT-REX of the University of Stuttgart is used [SBB+22]. The gamification approach, which this project followed, is a mascot development over a course, which was fed with points and newly acquired knowledge. This approach applied to the concept of this thesis integrates the gamification elements into the individual areas of the platform. The gamification elements, including the mascot adopted from IT-REX, are included in a low-code adapter, which can be integrated into the e-learning platform by lecturers of a course. To provide this adapter with meta-information, combined functionalities are unlocked with other installed adapters, and information about videos viewed and tasks solved by students are exchanged. Thus, the gamification extends through all installed adapters and results in a well-integrated gamification system across the whole course within the e-learning platform.

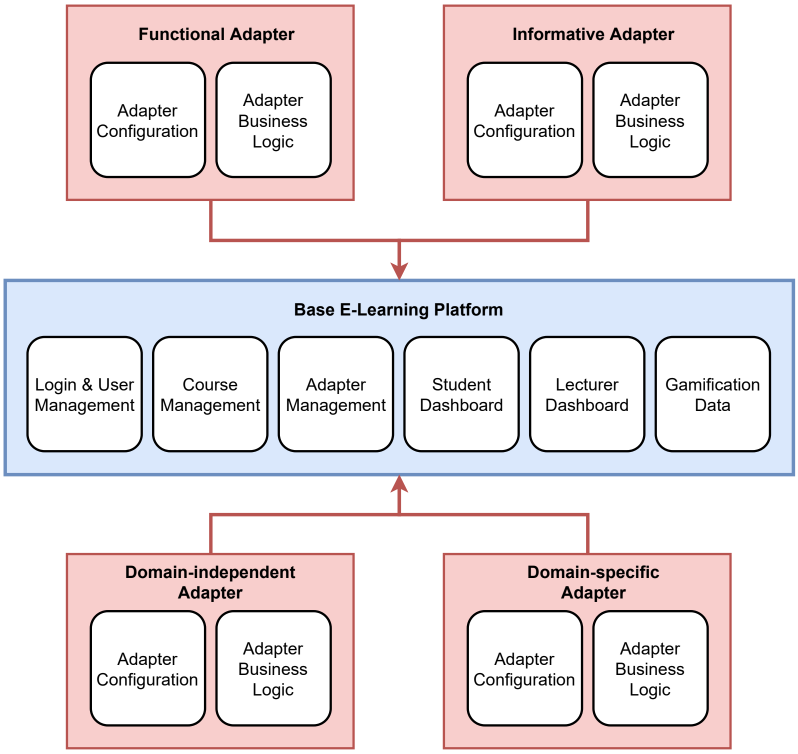

Figure 3.5: Layout and responsibilities of the base platform and different types of adapters.

3.4 Aspects of the Concept

This section describes the main features of this thesis concept. First, Section 3.4.1 outlines the aspects of the base e-learning platform. Then, the scope of the low-code adapters is defined in Section 3.4.2, followed by the interfaces and the data of the individual components in Section 3.4.3. A suitable implementation with a microservices architecture is also presented in this section. Section 3.4.4 provides various Business Process Model and Notation (BPMN) diagrams showing how the platform is used and which processes are performed within the LMS. Finally, Section 3.4.5 deals with the topic of lecturer support, how they can best handle the e-learning concept, and learn how to use it.

3.4.1 Base E-Learning Platform

This section deals with the concept of the base e-learning platform and which functionalities it provides for lecturers and students. In the following, several aspects of the platform are described and integrated into the context of this thesis. The section also explains how the gamification aspect is integrated into the platform and how the low-code adapters are embedded. The layout of the e-learning platform is shown in Figure 3.5.

Components of the Base Platform

The base platform consists of the essential components of a LMS. A part of the essential components are presented in Figure 3.5, but the included functionalities are extensible, depending on which base functionalities are needed for the LMS. The most important components are shown in the figure. All typical components such as login interface, user settings, and similar, as well as an appealing UI, are taken for granted and do not need to be explained in more detail within this context. Furthermore, the base platform provides course functionalities, such as course creation by lecturers. In addition to creating, managing, and removing courses, these courses can be published and made accessible to the students, enabling them to join them. Within a course, both students and lecturers are provided with a dashboard of functionalities, which are described in more detail in the following sections. In case of a need for customization by lecturers, they can access the course configuration. On this configuration page, they cannot only change the default settings of a course but also have access to a list of installed and uninstalled low-code adapters. The base platform receives this information about the availability of the respective adapters and displays them in the course configuration. Furthermore, the base platform offers numerous support options for lecturers to understand how to use the platform and request support for creating a course and educational material. It also provides the ability to submit feature requests to the development team of the LMS directly within the platform. A detailed description of the support capabilities of the platform is described in Section 3.4.5.

Student Dashboard

The student dashboard serves multiple purposes. First, it is intended to show students at a glance where they can access content such as educational material and what opportunities they have to learn in the course. Moreover, a well-structured home page for students has the advantage that they are more motivated to continue studying. The student dashboard comprises the adapters installed in the course, displays information, and offers options for direct use of adapter functionalities. The base e-learning platform obtains the required data from the individual adapters, aggregates the information, and displays it in the dashboard. The lecturers can customize the dashboard, allowing them to specify what the students' home page of the course looks like. Not only the layout of adapters and information but also the type and form of the presentation can be customized by the lecturers according to their preferences. Since the dashboard should not be the only way for students to use the functionalities in a course, there is also the option for students to access individual adapters directly. For this purpose, the adapters are presented like small applications in a quickstart area on the home page.

Lecturer Dashboard

The lecturer dashboard is analogous to the student dashboard, the home page of a course for the lecturers, and contains information and functionalities for the organization of the course. One area of the dashboard shows the options where lecturers can upload or create content for students. This ensures that lecturers can design courses and create educational material as efficiently as possible. Furthermore, general adapter functionalities and information are displayed in the lecturers' dashboard. The base platform obtains these functionalities and information from the installed adapters within the course. In addition, the lecturers' dashboard contains comprehensive analytics of the student's learning behavior and statistics. However, since this is not the focus of this thesis, it will not be addressed further. The dashboard also lists all the installed adapters in the course so that the lecturer has an orientation on the course's functionalities.

Gamification
Gamification elements are essential in a modern LMS. Based on discussions with vocational and technical pedagogy experts, three pillars were defined that the platform has to fulfill from a gamification perspective to achieve the desired benefits of gamification. These pillars are mechanics, aesthetics, and storytelling. Due to the modular approach of the concept, it was decided against a cross-platform integration of gamification elements. Lecturers have to decide for themselves whether they want to use gamification in a course or not. When lecturers decide to use gamification and integrate the corresponding low-code adapter into their course, this impacts the entire course. Each educational material will be linked to the corresponding gamification elements. The mechanics and storytelling are captured to some extent by other low-code adapters, as these techniques can only be applied within the adapters. Aesthetics, however, is also mainly due to the implementation of the UI and the presentation of educational material in a way that encourages learning.

Integration of Adapter Features
The low-code adapters are installed into the base platform to provide advanced functionality. The adapters are installed within a course so that the scope of the course delimits the adapters as such. The features of the adapters are integrated into the predefined structures of the base platform and do not fundamentally change them. These low-code adapters have their application area in which they are used and thus constitute an independent application within the platform. Within this area, the adapter is not restricted in its use; even the UI elements are deployed through the adapters. This allows the use of many adapters within a course without affecting the overall structure of the platform. Once installed in the base platform, the functionalities of the adapters can be filled with educational material by the lecturers and accessed and used by the students. In addition, when the adapters are integrated into the platform, information is retrieved by the base platform to fill the student and lecturer dashboard and to display an adapter overview page.

3.4.2 Low-Code Adapter

This section focuses on low-code adapters. Different types of adapters are examined, and the application areas are presented. Figure 3.6 shows how a set of example adapters is divided into different types to distinguish the categories. This section will show the benefits of the loosely coupled adapters for the platform and, ultimately, for the lecturers and students. The section also describes the possibilities of linking adapters to enable combined functionalities.

Functional Adapters
Functional low-code adapters contain functionality that students and lecturers can interact with. For instance, lecturers can create or upload questions in a quiz adapter, and students can answer them. When an adapter is functional, it has implications for the entire platform.
In the student and lecturer dashboards described in Section 3.4.1, information is presented and functions for direct work on educational material on the home page. Functional adapters can provide these functions to the base platform to integrate them into the corresponding dashboard. From a lecturer's point of view, functional adapters are classified as interactive when lecturers have to do something actively to make the adapter fully work.

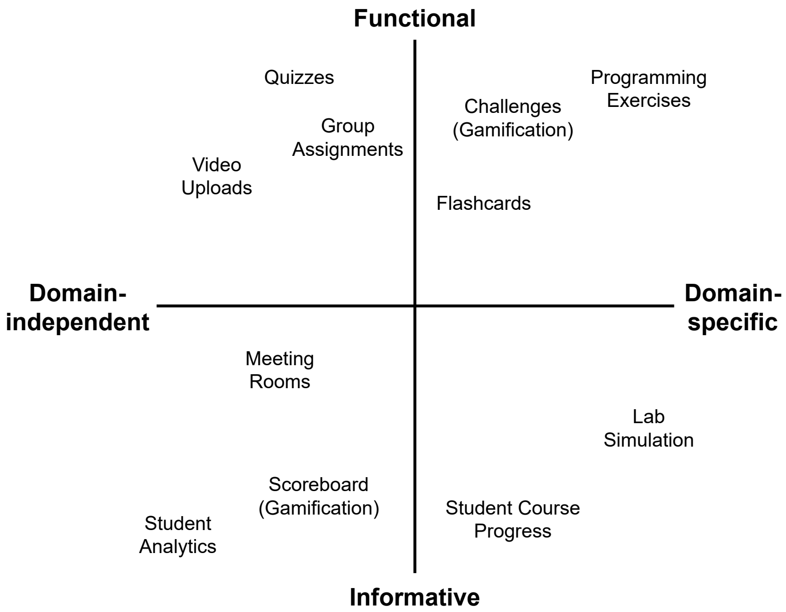

Figure 3.6: Categorization of a set of example adapters.

Informative Adapters

Informative low-code adapters exclusively comprise information presented in a variety of forms to both students and lecturers. The adapter can gather information independently and aggregate and present information from other adapters. For instance, this can be an adapter with student-specific learning progress. The learning progress is displayed to the student, but the student cannot interact with the adapter. Lecturers cannot interact with these adapters either. The adapters are for informational purposes only. For lecturers, informative adapters reflect the performance of students in a course. Since many adapters collect student information, it can be presented to lecturers as analytics. Also, lecturers can configure what kind of information from informative adapters should be presented to students.

Domain-independent Adapters

Domain-independent low-code adapters can be integrated into courses in various fields of study. The adapters have not been developed for a specific field of application but rather extend the e-learning platform with additional general features for everyone. Examples of these adapters are video uploads and quizzes. They allow lecturers to upload videos or create quizzes, and any lecturer can utilize these domain-independent adapters. The adapters do not specify the type of videos or quizzes that must be created or the study program in which they are taught. Domain-independent adapters have the great advantage that they only need to be developed once by the platform's development team, and then all lecturers can use them. Many customers can use such adapters, so their development is worthwhile.

Domain-specific Adapters
Unlike domain-independent low-code adapters, domain-specific adapters are developed for a specific application area within a study program. Therefore, the adapters depend on the study program and, in some cases, on the specific course. Examples of such adapters are, among others, flashcards as well as programming assignments. These adapters have an already assigned application area. They can be installed in the course of another study program, but this would barely be applicable.
The development of domain-specific adapters must be weighed against costs and benefits, as development by the development team can be time-consuming, depending on the requirements. Since these adapters only have users from one field of study, the development of domain-specific adapters must be carefully considered. However, the low-code adapters do not have to be developed entirely by the development team. External functionalities are also integrated into independent adapters. Thus, specific applications can be integrated and used as adapters in the platform. By extending domain-specific adapters with external applications, the development effort is reduced. Such adapters can, for instance, also use functions from *ArTEMiS* [KS18] to evaluate source codes.

Combined Features
Low-code adapters in the e-learning platform are not stand-alone and are not necessarily only responsible for specific functionalities. Fundamentally, all adapters have their functionality but can extend it if other compatible low-code adapters are installed in the course. Compatible adapters installed in a course can then be linked together so that they communicate with each other and exchange information and functionalities. For instance, the adapter with a video function could be linked to the adapter with a chat function, creating the combined feature that students can now chat while watching a video together. Combined features help design the modular concept so that all adapters interact with each other. Ultimately, it is no longer evident which functionality is provided by which adapter.

3.4.3 Interfaces and Data

This section examines the interfaces and the data exchange between low-code adapters and the base e-learning platform. Not only the interfaces between the adapters and the platform are outlined, but also the interaction between adapters. First, a suitable architecture focusing on the interfaces is explained, followed by a data model for interfaces between adapters. The concept's scope is designed on a meta-level, so the individual components are not limited to a specific technology but can be implemented variably.

Figure 3.7 shows a suitable architecture of the e-learning platform with low-code adapters. The base platform has an *1-to-n* relationship with the adapters, whereby at least one adapter must be integrated into the platform since the base platform has no functions other than course creation, as described above.

A Microservice architecture is indispensable for the concept since this enables easy scaling of the adapters if individual adapters need to be used more intensively [Kap22]. Similarly, if the LMS is accessed by many students simultaneously.
Furthermore, with the appropriate communication method, such as messaging, a microservice architecture enables loose coupling between the microservices of the base platform and the adapters connected to it.

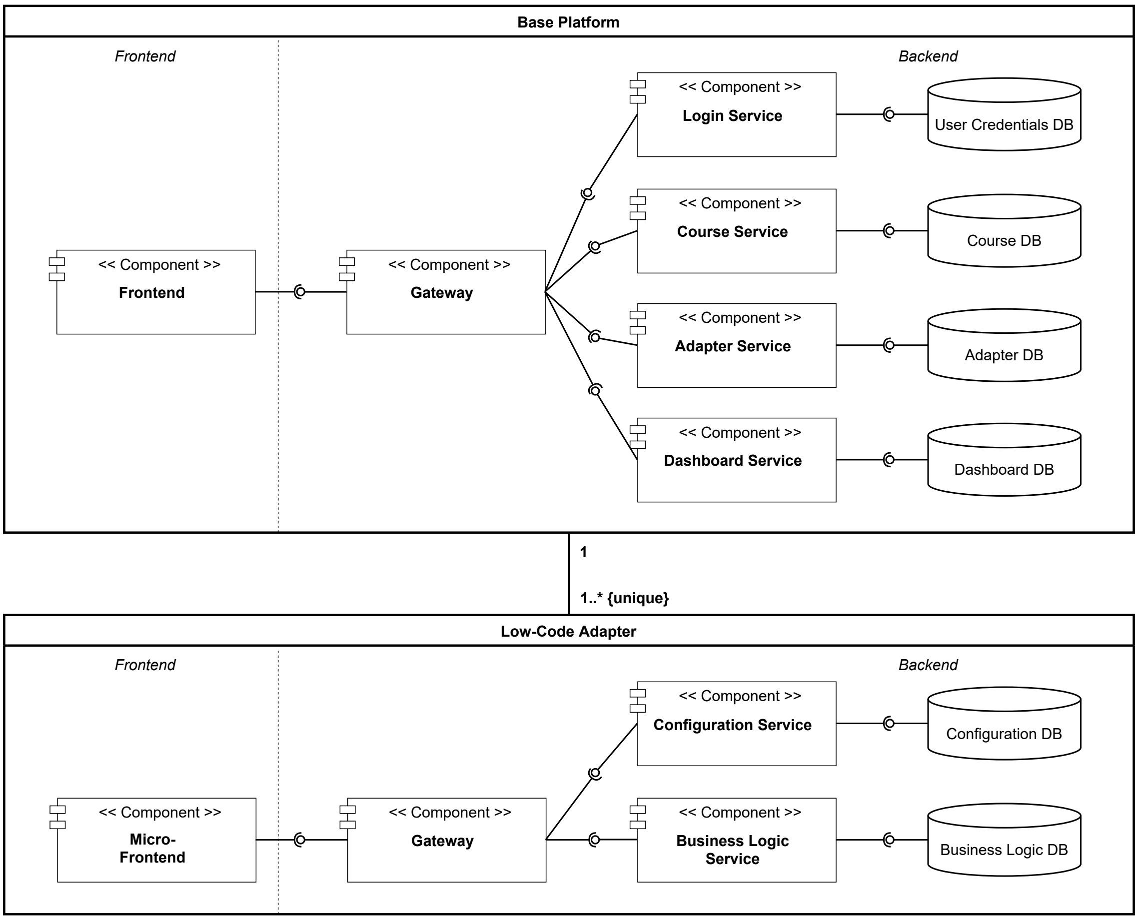

Figure 3.7: Concept of a suitable architecture with interfaces.

In Figure 3.7, the base platform is illustrated separately from the low-code adapters, which it functionally is. The software architecture follows the approach of the Application Programming Interface (API) gateway design pattern [Kap22]. The gateway shown is the same; thus, the adapters are bound to the same gateway as the platform itself.

The frontend of the base platform includes all elements of the platform's functions and leaves space for the integration of the adapters. Each adapter has a dedicated area in the platform to integrate functionalities and display them to the students. The content of these spaces is dynamically loaded by the micro-frontend elements of the individual adapters, allowing them to be customized and integrated into the platform at runtime. This enables flexible customization of the adapters, as they are dynamically loaded into the platform. It also simplifies the development of new adapters. This approach can be standardized by developing individual micro-frontends with the appropriate functionality rather than having to customize the frontend of the entire base platform for each extension.

The gateway provides the interface between the frontend and the backend. On the one hand, user input from the frontend is received and distributed to the responsible backend microservices. On the other hand, the information from these microservices is made available to the frontend to display in the UI. The gateway also serves as an interface between all microservices and, thus, as a connection between the base platform and the adapters and among the adapters themselves.

Since the concept focuses on the application of customizable low-code adapters in e-learning platforms, only the components required in the context of this thesis were included in the LMS. These include the microservices in the base platform: *login service*, *course service*, *adapter service*, and *dashboard service*.

The *login service* provides the platform with authentication and authorization capabilities so lecturers and students can log in and use the LMS accordingly. Only lecturers can make adjustments to courses and adapters. Accordingly, the login service contains a user information database, which is required for user identification and user-specific data.

The *course service* includes all course-specific functionalities, such as creating and managing courses. The courses created in the platform are stored in the corresponding database. Courses are one of the central components of the LMS, as they form the frame of the learning environment. Depending on the course, low-code adapters can be installed in it. Each course thus receives an individual set of configured adapters and functionalities.

The *adapter service* handles the management of the adapters in the platform. The service stores adapter information in its dedicated database. Whenever an adapter becomes available or breaks down, this is immediately reported to the adapter service, which then initiates the appropriate steps in the platform. The service is also involved in the installation and uninstallation of low-code adapters.

Lastly, the *dashboard service* is responsible for the composition of the dashboard for students. It stores dashboard configurations by the lecturer in its database. Information provided by the functionalities of the individual low-code adapters is collected by the service, aggregated, and presented to the students in a structured way. The dashboard service is in close data exchange with all adapters installed in a course.

Besides the micro-frontends and the gateway that connects the entire LMS, the low-code adapters consist of the following microservices: *configuration service* and *business logic service*. The business logic service represents the functionalities of the adapter. Depending on the scope and content of each adapter, the service can be divided by the developers into smaller, separate microservices. However, since the approach of this thesis provides a domain-independent platform, all functionalities of an adapter are united in the business logic service.

The *configuration service* contains all adapter-specific configurations stored in the configuration database. The configurations can be customized by the lecturer and are individual for each course. Thus, each course can have a different configuration of the same low-code adapter. The configuration service also communicates with the same microservice of other adapters linked together and synchronizes configurations of combined functionalities.

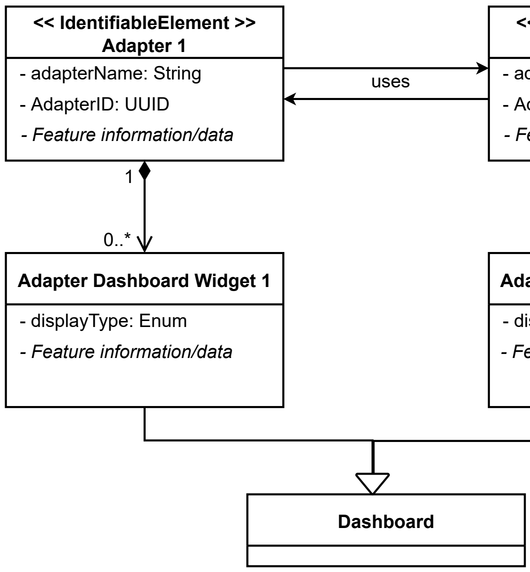

Figure 3.8: Dashboard data model as UML diagram.

The *business logic service* contains the complete functionality of the low-code adapter and has a corresponding database in which, for example, educational material and student progress information is stored. The service is customized by the configuration service and tailored to the needs of the lecturer. The information and functionalities of the service are displayed on the platform. When adapters are linked, functionalities are made available to other adapters so they can use these functions and exchange relevant data.

Figure 3.8 illustrates the exchange of data between adapters, which is displayed on the student dashboard. Thereby, the data exchange of an arbitrary number of low-code adapters, '*Adapter 1*' and '*Adapter n*', is shown. The adapters are linked to each other and accordingly use functionalities and data of each other. The adapters identify themselves by a unique identification number and the corresponding course affiliation. Within a course, each adapter might have widgets, which are assembled in the student dashboard. The widgets are related to the adapters in an *1-to-n* relation since an adapter can contain any number of widgets. The reason is that the lecturer has the flexibility to create new widgets and customize them according to the needs. However, in doing so, the lecturer is not forced to use widgets as the functionalities of the adapter can also be integrated into the e-learning platform independently of the student dashboard. The individual widgets access the corresponding adapter information. They can display this information and the adapter's functionality so that the students can interact with the adapter directly on the home page.

3.4.4 Business Processes

This section focuses more on integrating the low-code adapters and illustrates them in their processes as BPMN diagrams. In the following, the three most essential processes regarding the low-code adapters are described: course creation, installation and linking of adapters, and adjustments to installed adapters.

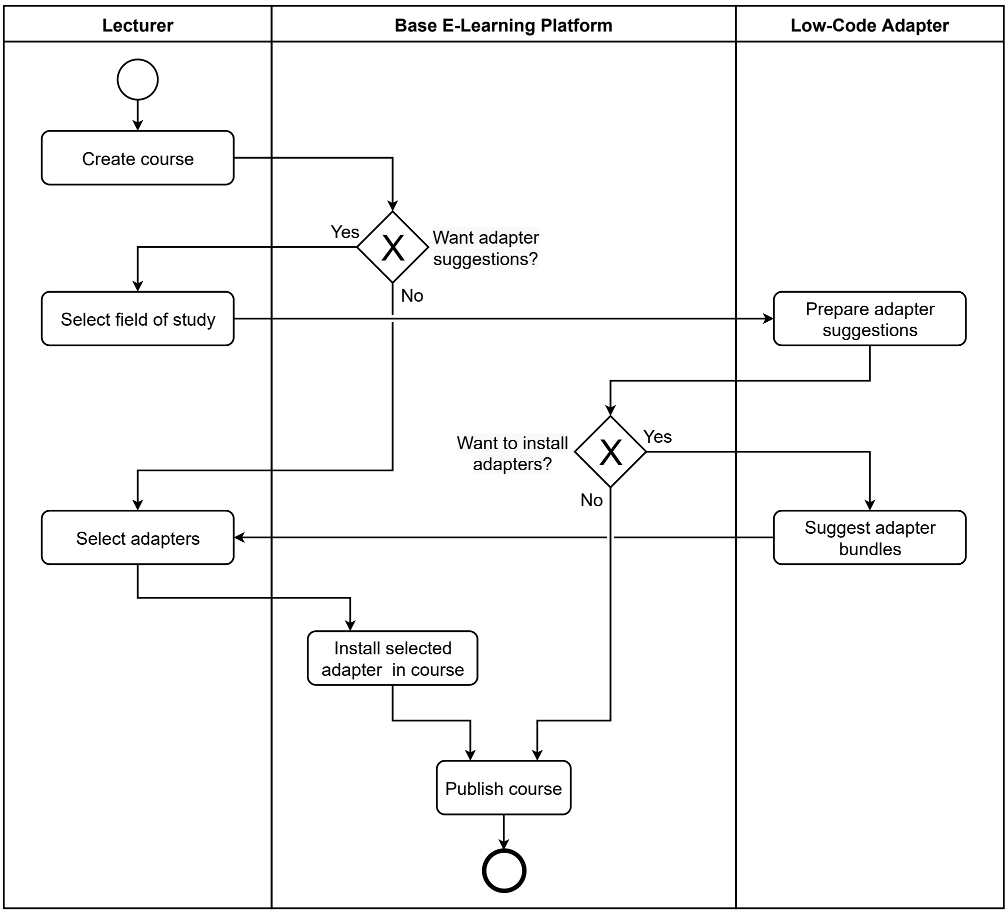

Figure 3.9: BPMN of course creation.

Figure 3.9 shows the BPMN for course creation within the platform from the perspective of the lecturer, the base platform, and the low-code adapter. For the course creation by the lecturers, some steps are required within the platform, while the lecturers only need to access the course creation page and select the low-code adapters they want to use for their courses. The e-learning platform offers lecturers to specify their fields of study during the course creation. Accordingly, the corresponding adapter suggestions are prepared from information provided by the low-code adapters. The lecturers can select from the adapter bundles suggested by the low-code adapters. Subsequently, the selected adapters are installed in the base platform, and the course is published, making it accessible to the students.

The process of installing and linking adapters is shown in Figure 3.10. The focus is on installing the adapters, as this is a sub-step in the course creation process. However, since the adapters can still be added, removed, and adjusted after the course has been created, the scope of this process is extensive. Not only the installation of a single adapter in the base platform is shown, but rather how several adapters can be installed and linked together. The lecturer's first step is selecting the adapters that should be installed in the course. It is the same process, whether it takes place during the course creation or afterward. The selected low-code adapters are installed in the base e-learning platform, and an initial setup is invoked simultaneously for each of the installed adapters.

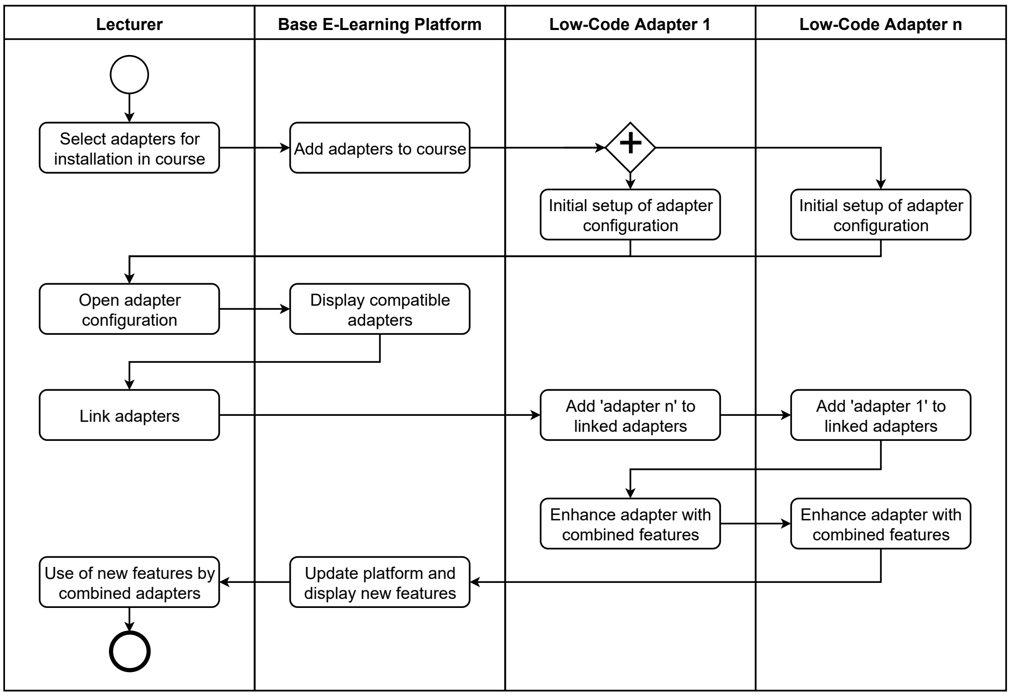

Figure 3.10: BPMN of adapter installation and linking.

The setup contains the default configuration of the adapter in a course. As shown in the BPMN in Figure 3.10, the two adapters '*Low-code Adapter 1*' and '*Low-code Adapter n*' are representative for any number of installed adapters. After successfully initializing the adapters, the lecturer can open the adapter configuration within the course to adjust the selected adapter. In addition to the configuration settings for the adapter, the platform also shows the compatible adapters that were also installed in the course. In this way, it is clear to the lecturer which low-code adapters can be linked together. If the lecturer wants to link one adapter with the other and thus obtain combined functionalities, the adapters communicate with each other and link internally. Both adapters extend their functionalities with the combined functionalities available through the linking. Subsequently, these functionalities and settings are updated in the platform and displayed to the lecturer, who can now use the new functionalities.

The BPMN in Figure 3.11 shows the adaptability of the adapters from the lecturer's perspective, the base platform, and the low-code adapters. For this purpose, the lecturers are given an easy way to make adjustments when they identify a new need for adjustment or enhancement. In the platform, the lecturer can view the course settings, which show general information about the course and the adapters installed in the course. The information about which adapters are installed in a course is conveyed by the adapters and aggregated and displayed in the platform. The lecturer can then delete individual adapters from this adapter list. If the lecturer decides to uninstall one or more adapters, the adapter is removed in the reverse direction of the installation process. The platform removes the adapter and the associated functionalities while the adapters unlink all interfaces and extensions in other adapters.

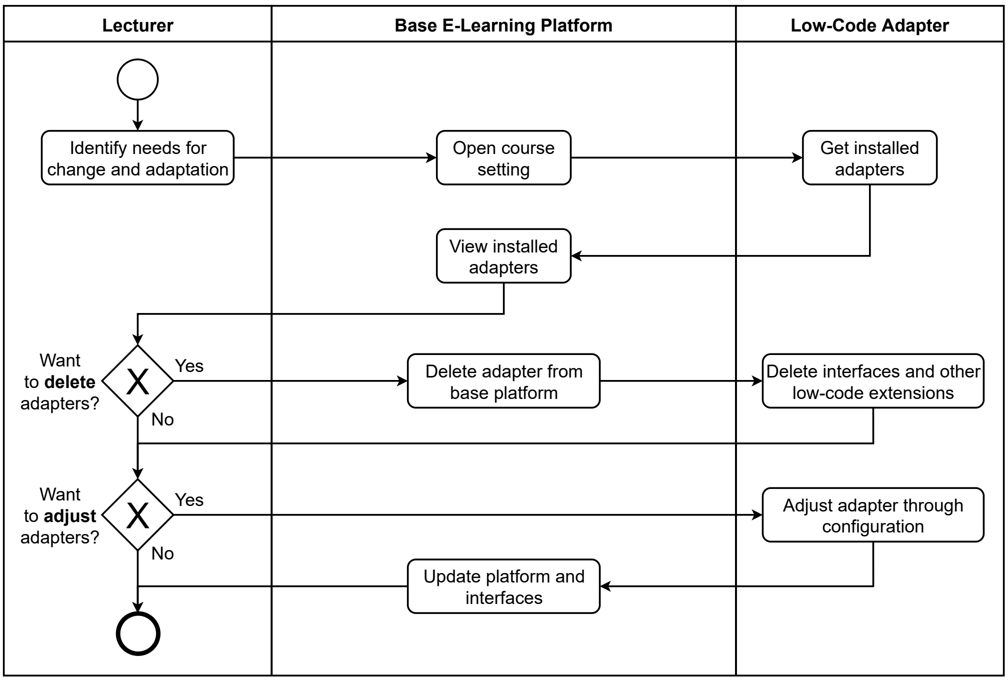

Figure 3.11: BPMN of adapter adjustments.

However, if the lecturers only want to make adjustments to individual adapters, they can modify and save the configurations of the individual low-code adapters directly in the respective adapter configurations. These are then applied to the platform, and both functionalities and changes in interfaces are applied.

3.4.5 Lecturer Support

As the e-learning concept introduces some changes and enhancements compared to established LMSs, lecturers need to be supported in dealing with the new platform. For this purpose, the concept includes some aspects in which lecturers can be supported. It may be assumed that the more IT background lecturers have, the easier for them to deal with the new platform.

Internal support
The platform's internal support consists of several levels. The top level of support is a platform-specific tutorial for lecturers, in which information about functionalities and views is displayed and explained to them with an overlay on the platform. This tutorial is visible in the entire platform when used for the first time but can also be skipped or redone later. The tutorial overlay also includes hints for quickly designing and customizing a course and installing low-code adapters.

In addition to the tutorials, mouse-over information and help areas are integrated, which explain the current page and the actions that lecturers can perform on this page. Especially for the functionalities of individual adapters and how they can be integrated into the platform must be explained in detail.

Much information and support must be provided on the adapter level since especially adapters used by lecturers for the first time can be more difficult to understand. For this purpose, adapter overview pages are provided, on which a wide variety of information regarding the adapters is presented. On the one hand, the functions of the adapter itself have to be described in detail to offer these functionalities to the lecturer and to present their possibilities. In addition, this overview page must include possible configuration settings to give lecturers insight into how they can tailor the adapter to their needs. Finally, it must also be shown on the overview page with which other low-code adapters the selected adapter can be linked and which combined functionalities can be achieved as a result.

Finally, a course overview page should display an outline of the adapters installed in a course and illustrate them as a diagram if support is required. The adapters installed in the course are displayed as nodes connected to the edges. These show the dependencies among themselves and which adapters communicate with each other. A diagram makes it easier to understand which components make up a course in detail.

The e-learning platform should also automatically recognize when a lecturer has not worked with it for a certain period and thus automatically display more information and support to get the lecturer to work with the platform more frequently.

External support
As external support, a support team is required to assist the lecturers with questions not answered by the platform. The support team is responsible for supporting the lecturers in their course creation and advising and guiding them in selecting the appropriate adapters and the corresponding configuration. These support teams can consist of students or other staff members familiar with the platform. In case of questions and need for support, lecturers can request the necessary support via the corresponding functionality in the platform.

The support team is the first point of contact when a feature request is needed. If lecturers have a requirement not covered by corresponding adapters, the support team must request further development of the platform from the responsible development team. This development team is not only responsible for the development of the e-learning platform but also for the further enhancement of the low-code adapters and the adjustments as well as the control of the interfaces between the adapters. Since the lecturers mainly have no technical experience, a central development team is essential to enhance the platform.

3.5 Domain Model as UML

To provide a general overview of the required components and the concept's scope, the domain model in Figure 3.12 shows the components as a UML diagram. The central and most important components comprise *(1)* Course, *(2)* Adapter, and *(3)* Dashboard.

(1) The course element involves the creation of a course by a lecturer and the participation of students. The course includes information such as the course name and a course description, which are identified as minimal information and remain unconstrained. Thereby, the course to students has an *1-to-n* relationship since any amount of students can participate in a course.

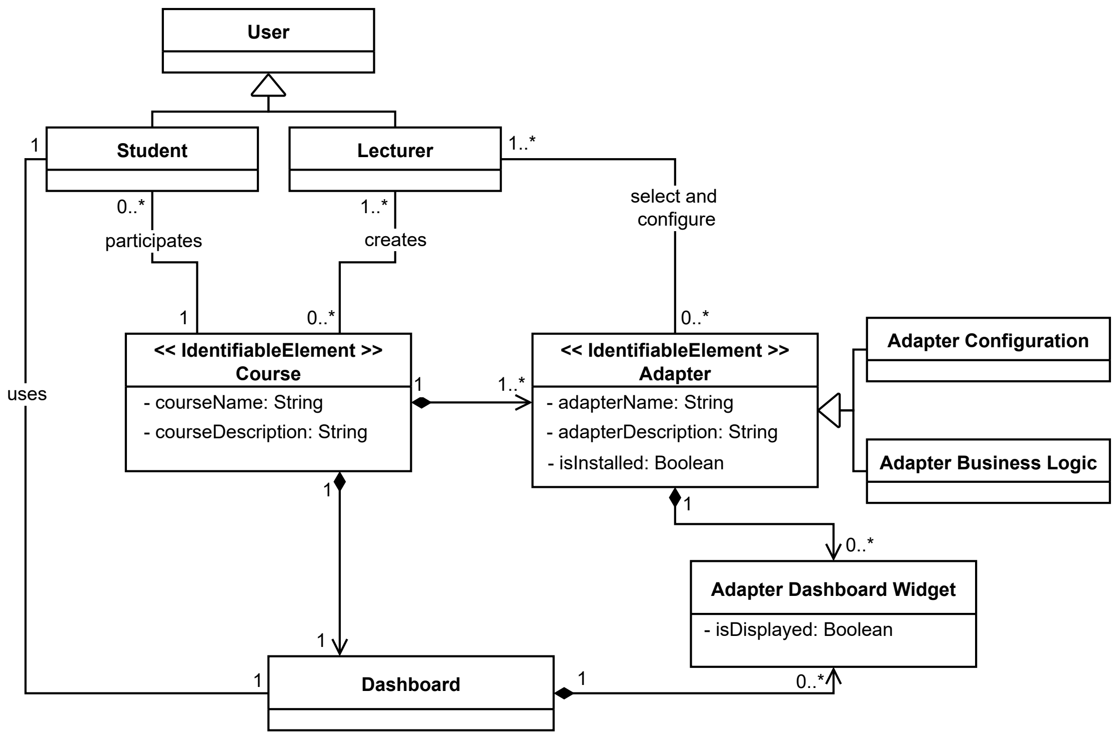

Figure 3.12: E-Learning platform domain model.

On the other hand, lecturers have a *n-to-m* relationship with the course since one lecturer can create multiple courses, and multiple lecturers can collaborate on one course. Therefore, the components illustrated in the figure are established in most LMSs and do not include extensions through the content of the elaborated concept.

(2) The course is related to the adapter in the relationship *1-to-n* so that a course contains at least one adapter. This allows any amount of adapters to be installed in a course. The adapter also contains information regarding the adapter name, the description of the functionality, and whether it has been installed in a course. These are the minimum required information and can be extended beyond the concept. The adapter is composed of the adapter configuration and the adapter business logic. The configuration includes all specific adapter settings. The adapter configuration also includes the settings of the combined functionalities with other adapters. The adapter business logic includes all adapter-specific functions and can, therefore, also vary greatly depending on the adapter. The lecturer has a *n-to-m* relationship with the adapter since the lecturers who have created a course and collaborate in it can select and configure all adapters within a course.

(3) The dashboard exists once in a course and displays the course's main content on the student's home page. A dashboard is adapted to a course and a single student since the student gets personal and individualized educational material displayed on it. An adapter can contain several dashboard widgets that can display different functionalities and information. The dashboard is formed from different widgets of adapters and thus has an *1-to-n* relationship to the dashboard widgets.

4 Implementation

This chapter outlines the implementation of the previously described concept of this thesis. First, Section 4.1 gives an overview of the main aspects like functionalities and features of the prototype and the architecture of the implementation. This includes a description of the considerations of which approach was taken to implement the concept in the prototype. Then, in Section 4.2, the backend and frontend of the prototype are outlined, and UI mock-ups are shown to envision how a LMS with low-code adapters could look like. Finally, Section 4.3 explains the tools and technologies used to implement the prototype. The complete documentation of the low-fidelity prototype can be accessed on *Zenodo*[1].

4.1 Overview of the Main Features and Architecture

This section describes the main features of the low-fidelity prototype for the implementation of the concept and the components required for it. The architecture of the implementation is then sketched and described. Finally, it is outlined which components must work together to obtain a Minimum Viable Product (MVP) that can be evaluated based on the previously defined requirements.

The low-fidelity prototype aims to evaluate the concept for its feasibility and thereby examine its advantages and disadvantages. Therefore, the prototype focuses primarily on implementing parts of the functionality that are sufficient for an initial evaluation and not on implementing a whole LMS with all functionalities. The focus is on the implementation and realization of the features due to the set of requirements and needs of the stakeholders. Therefore, the architecture of the previously described concept contains more functionalities than the prototype depicts at this stage. The main focus of the implementation is the interaction of the base platform with the installable low-code adapters. The user should be able to get an impression of the scope in which the platform can be used and what benefits this implies. Lecturers should be able to experience and try out the entire process of the e-learning platform, from creating a course to providing educational material. This includes the installation of low-code adapters in the platform for specific desired functionalities. On the other hand, students are empowered to use this prototype to get an idea of what the platform will look like from their perspective and how they can use it. To get a Proof of Concept (PoC), much further information is mocked up in the implementation. Only the main aspects of the concept are implemented in more detail. To develop the low-code adapters required for the evaluation and to cover a wide range of application areas, the prototype focused on developing the following adapters: *Gamification*, *Flashcards*, *Matching Exercise*, *Course Timeline*, *Knowledge Progress*, *Video Upload*, and *Quiz Upload*. The adapters and their relations to each other are also shown in Figure 4.4 and will be described in more detail in a subsequent section.

[1] https://doi.org/10.5281/zenodo.7065141

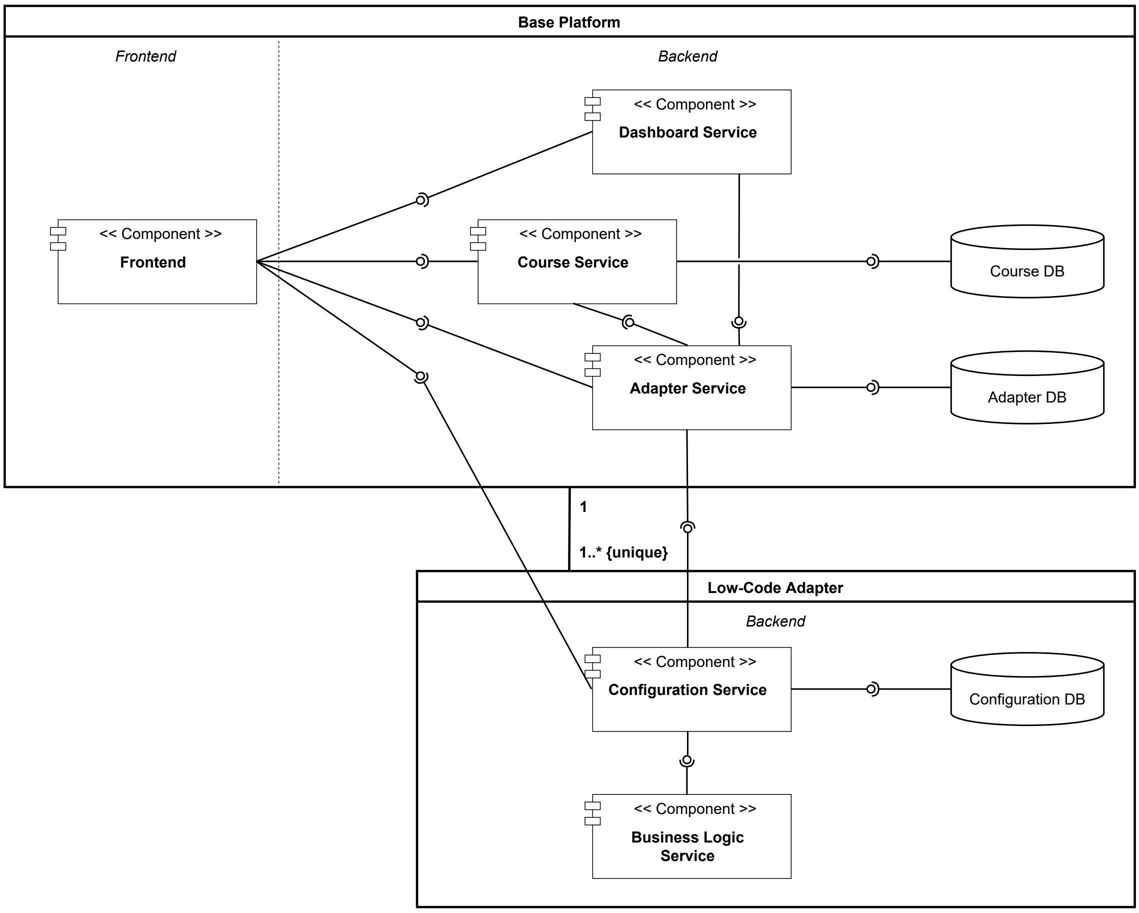

Figure 4.1: High-level overview of the prototype architecture.

An essential prerequisite for implementing the prototype was a high degree of modularity. The modular coupled low-code adapters also contribute to this. Thus, a microservice software architecture was deliberately chosen to be able to separate functionalities and adapt them separately. This promotes not only good maintenance of the prototype but also an easy extension in the future. In addition, the API logic was separated from the business logic to obtain a more precise control flow.

Figure 4.1 shows an overview of the prototype architecture. As described in the concept, the prototype is divided into a base platform and low-code adapters, each containing microservices with corresponding functionality. The implementation was limited to the bare essentials so that the backend of the base platform only consisted of a dashboard, a course, and an adapter service. Other services, such as a login service, are not used and implemented in the low-fidelity prototype, as they are outside the focus of the development. In addition, the implementation contains a database for courses and adapters. On the other hand, the base platform contains a frontend, which communicates with the services in the backend via interfaces. Due to the timeframe of the thesis, a dedicated gateway was not used in the implementation of the prototype.

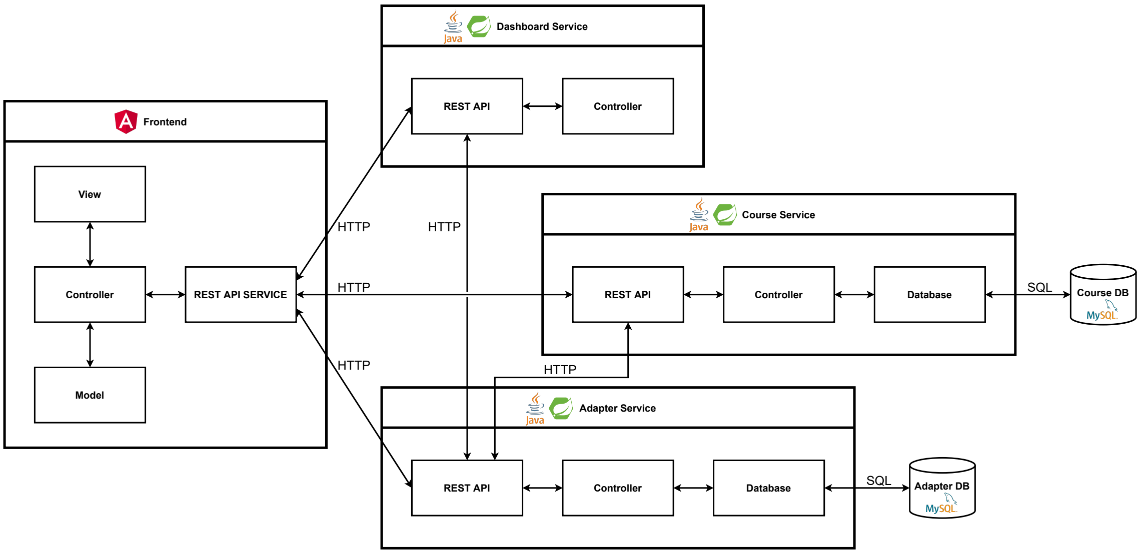

Figure 4.2: Interfaces between the frontend and the base platform microservices.

Besides the base platform, it is possible to connect unlimited low-code adapters to the platform. In the context of the prototype, the adapters use the frontend elements from the base platform but can later be integrated by micro-frontends, as described earlier in the concept. The adapters consist of two microservices, the configuration and the business logic service. The business logic service has already been created in the source code of the prototype. However, it does not include any functionalities since the focus of the prototype is on the composition of adapters and not on the functionality of individual adapters. The configuration service contains a configuration database. In the prototype, the low-code adapters' configuration service communicates with the base platform's adapter service and the frontend via interfaces. A gateway will control this communication at a later stage.

Figure 4.2 illustrates the interfaces used by the frontend of the prototype to communicate with the microservices of the base platform. The architecture of the frontend is designed based on the standard Model View Controller (MVC) pattern and built with the *Angular* project structure. For the backend and frontend communication, Hypertext Transfer Protocol (HTTP) requests are used. The frontend receives data that is presented to the user via the Representational State Transfer (REST) API service, which sends HTTP requests to the individual microservices in the backend. The controller processes this received data and prepares it for visualization, and matches the received data with the logical data structure contained in the model. Subsequently, this information is displayed in the view and thus to the user in the UI. On the other hand, the REST APIs of the microservices receive the HTTP requests of the frontend and process the data within the controller. In addition, *MySQL* databases for storing courses and adapters are integrated in the course and adapter service. These microservices also require an interface for processing and obtaining data from the database. The *MySQL* databases are implemented so that any SQL database can be used. Thus, *PostgreSQL* can also be used and easily exchanged. The data retrieved from or entered into the database is controlled via Structured Query Language (SQL) commands and returns information to the controller. In addition to the communication between the frontend and backend, the microservices also communicate with each other to a large extent. To achieve this, they use the existing REST API interfaces of the respective microservices, which they address to obtain the required data.

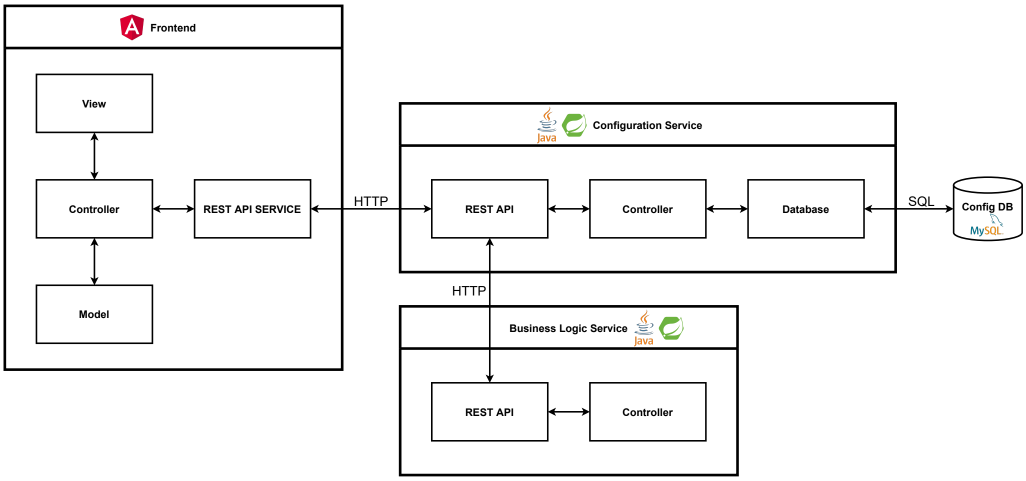

Figure 4.3: Interfaces between the frontend and the adapter microservices.

The communication type was chosen for simplicity and experience with REST interfaces. In more advanced implementations, messaging should be chosen as an asynchronous communication type for a loose coupling between the individual microservices. The implementation with REST is sufficient for the intended use of this low-fidelity prototype.

The interfaces between the frontend and the microservices of the low-code adapters are shown in Figure 4.3. As described earlier with the base platform microservices, the frontend uses the REST API service to interact with the individual adapters. The focus of this implementation is the configuration service, which manages the settings and configuration of the adapter. The settings are stored in a configuration database and are course related. A minor part of this elaboration is the business logic service, which, depending on the type and scope of the adapter, either covers the entire functionality of the adapter or can be subdivided into further tailored microservices. Since the business logic service makes up a small part of the implementation and the microservice for each adapter was stored in the code repository without much functionality, it is not crucial for further consideration of the prototype.

As described before, a wide range of different adapters was identified for the implementation and use of the prototype. Figure 4.4 describes these adapters and how they interact. As shown in the figure, all adapters are initially linked to the base platform. When adapters are installed in a course, they can communicate directly with each other. The defined compatibility was based on the collection of synergies between individual functionalities. Since not every functionality has commonalities with another or can be combined, it had to be clearly defined before the development of the prototype, which possibilities of combination may exist. Possible combinations and synergies can also be added later and anchored in the system. However, in the context of the prototype, it is relevant to show that adapters can share functionalities, complement each other and create standard functionalities and benefits, regardless of whether these functionalities are needed in the context of the prototype.

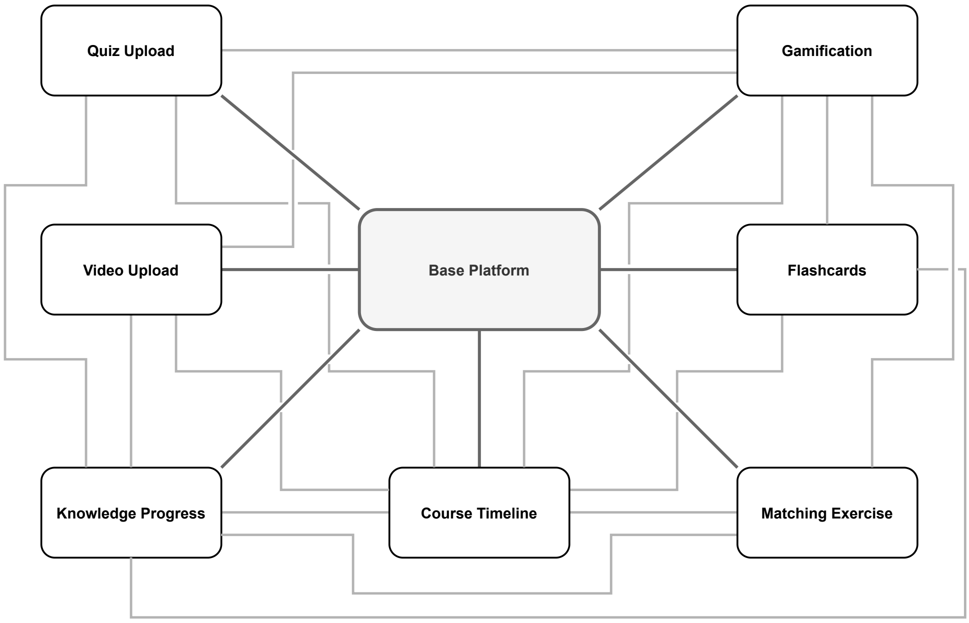

Figure 4.4: Overview of the low-code adapters and their mutual connections.

The adapters covered in the prototype simulate the following functionalities and compatible adapters:

Gamification

- This adapter provides a gamification system for students to be more motivated to attend classes and listen to lectures and complete assignments.
- Compatible adapters: *Quiz Upload*, *Video Upload*, *Course Timeline*, *Matching Exercise*, *Flashcards*

Flashcards

- This adapter provides students with a repetitive learning style, meaning they must learn and repeat previously learned vocabulary and educational material to improve in the course.
- Compatible adapters: *Knowledge Progress*, *Course Timeline*, *Gamification*

Matching Exercise

- This adapter provides students with a learning task in which they have to match vocabulary words and their translations, thus improving their own vocabulary.
- Compatible adapters: *Knowledge Progress*, *Course Timeline*, *Gamification*

Course Timeline

- This adapter provides the ability to structure the course and schedule and organize uploaded content.
- Compatible adapters: *Quiz Upload*, *Video Upload*, *Knowledge Progress*, *Course Timeline*, *Matching Exercise*, *Flashcards*, *Gamification*

Knowledge Progress

- This adapter provides students with specific statistics about their performance in the course, as well as indications of areas in which they can improve.
- Compatible adapters: *Quiz Upload*, *Video Upload*, *Course Timeline*, *Matching Exercise*, *Flashcards*

Video Upload

- This adapter provides the possibility to upload any type of videos.
- Compatible adapters: *Knowledge Progress*, *Course Timeline*, *Gamification*

Quiz Upload

- This adapter provides the possibility to upload any type of quiz.
- Compatible adapters: *Knowledge Progress*, *Course Timeline*, *Gamification*

4.2 Low-Fidelity Prototype Implementation

This section deals with the implementation of the low-fidelity prototype. First, the implementation of the backend and then the frontend is described. Since the backend is divided into the base platform and the low-code adapters, the backend implementation for the base platform is explained first, followed by the adapters. Due to the timeframe of this thesis, both the frontend and the backend were not fully implemented, and some content was mocked up and created as placeholders. Since the entire design of the prototype was needed for the evaluation, the mock-ups were used to ensure a clean user experience. The source code for both, the frontend and backend can be found on *Zenodo*[2] and *GitHub*[3].

The base platform's backend contains the platform's business logic. It allows the user to create and manage courses and select and install available low-code adapters. It also includes the dashboard functionality needed for students, in which information from used adapters is prepared and displayed on the students' front page. The scope of the implementation of each microservice in the base platform extends as follows:

[2] https://doi.org/10.5281/zenodo.7064378

[3] https://github.com/Nimeggis/MA-prototype

Course Service
The course service includes all functionalities necessary for course management within the prototype. The course creation parameters received from the frontend by a lecturer are processed, and a new course is created in the course database with these parameters. Then, by calling the course, the information can be retrieved. Course-specific data is entered in the database, such as the list of low-code adapters installed in the course. If the lecturer adjusts these, the course service makes the necessary adjustments to the corresponding data in the database. When a course is deleted, all data associated with the course is removed from the database.

Adapter Service
The adapter service contains all functionalities in the implementation, which are connected to the management of the adapters. The adapter database contains an overview of which adapters are available and what information or features they bring. Information is stored on which functionality the adapter brings and which other adapters are compatible.
The structure of the previously described possibilities of interaction between adapters is primarily defined here and is subsequently used by the adapters. In addition, it is stored for each adapter, whether it is an informative or functional adapter. Thus the lecturer has to upload and embed material or whether an adapter extends the platform's functionalities without the lecturer having to provide additional material. Thus, the adapter service is used as soon as either functionalities or information about the adapters are requested and lecturers want to install and use them in the platform.

Dashboard Service
The dashboard service is responsible for preparing information from individual adapters and the subsequent presentation on the dashboard. Therefore, the service does not contain a database since information is obtained exclusively from the course service and the individual adapters. In the implementation and accompanying mock-ups of the functionality of adapters, less emphasis is placed on the presentation of information but rather on which adapters are installed in a course, which adapters are linked together, and whether the lecturer has enabled an adapter to be displayed on the student's dashboard. The backend of the low-code adapters contains the business logic of the adapters. It allows the user to be able to use the functionality of the adapter, to be able to adjust configurations independently and to be able to connect adapters. The scope of the implementation of each microservice in the low-code adapters extends as follows:

Configuration Service
The configuration service in the low-code adapters covers the entire configuration functionality of the individual adapters. The configuration of an adapter is course-specific so that different configurations of an adapter are used in different courses. In the configuration database, the configuration is divided into adapter-independent and adapter-specific. Each adapter's configuration stores which other adapters it has been linked to. In addition, it must be stored in the configuration whether the adapter should display information on the student dashboard. Besides the information required across all adapters in the same way, each adapter must store its configuration settings. These are different in each adapter's configuration database. All information and settings the lecturer adjusts are processed by the configuration service and modified accordingly in the database. At the same time, when an adapter is called, the information is loaded from the database and made available via the configuration service.

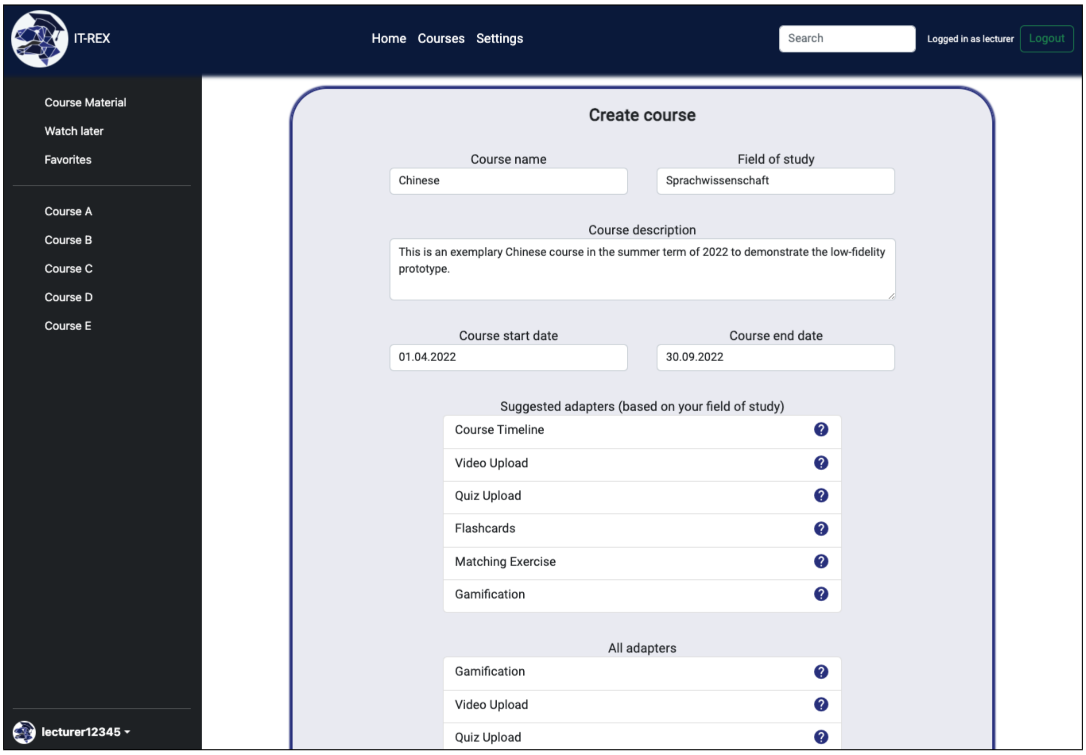

Figure 4.5: Implementation of course creation functions.

Additionally, the configuration service must communicate with other adapters to link them together. Suppose the lecturer links one adapter to another. In that case, this must not only be stored in the configuration database of one adapter but also address the other adapter to be linked.

Business Logic Service

As described above, the business logic service contains the functionality of the individual adapters. In the context of this prototype, the service has already been created and stored in the repository but does not yet contain any information or functionality in any adapter.

The frontend implementation uses the functionality provided by the backend and mocks up the functionality required for a good user experience. Additional features are also mocked up for a good-looking UI. For instance, the login functionality with user management is mocked up, while the course creation and modification use the backend functionality. Thus, the implementation of the e-learning platform has been limited to the most essential, so all functionalities that are not part of this elaboration have been mocked up. As shown in Figure 4.5, all displayed e-learning elements such as the sidebar, navbar, and user information are mocked up, and only the functionalities that were elaborated in the context of this thesis are implemented in greater detail.

Figure 4.5 shows the implementation of the frontend functionality for creating a new course by the lecturer. In addition to being able to fill in basic course information, the lecturer is also offered the option of specifying the field of study. As already described in the concept, suitable low-code adapters are suggested to the lecturer by selecting the field of study.

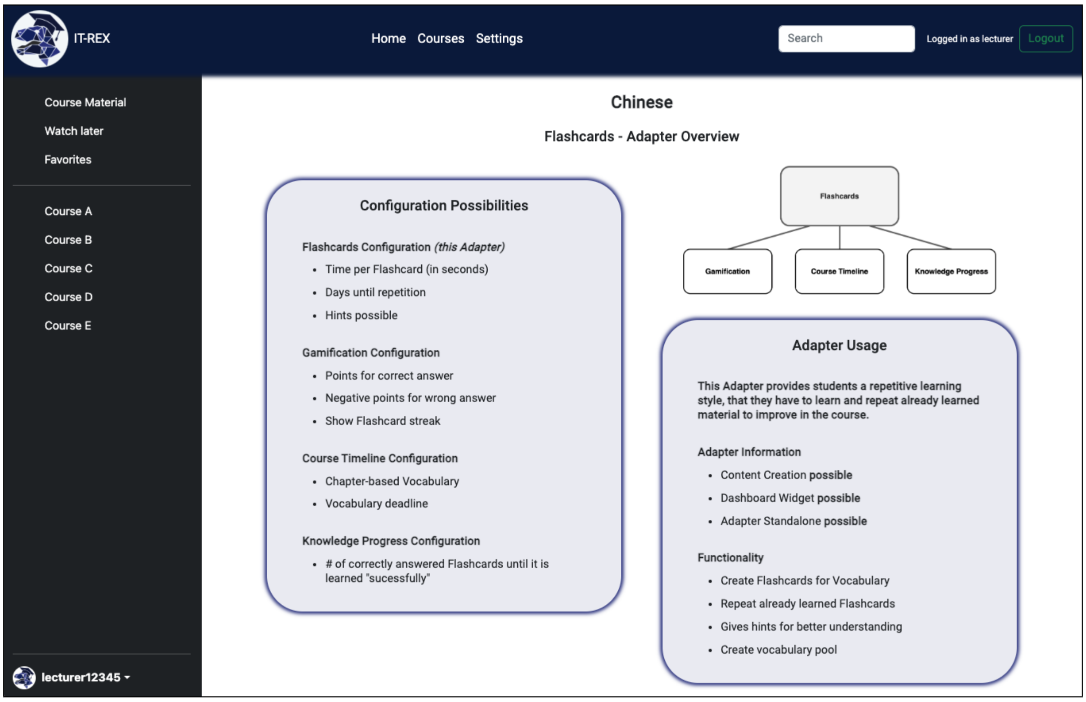

Figure 4.6: Implementation of the adapter overview functionality.

Essentially, the lecturer can make an initial selection here as to which functionalities are essential for the course and are needed for teaching. Furthermore, the lecturer gets support in understanding the adapters by clicking on the help of the respective adapters. Finally, the lecturer will be redirected to the adapter information page, which is shown in Figure 4.6.

The adapter overview of each adapter, shown in Figure 4.6, gives the lecturer an overview of an adapter and how it integrates into the base platform, as well as which other adapters it can interact with. The implementation within the prototype shows, on the one hand, the configuration options of the standalone adapter and the configuration options when this adapter interacts with others and enables standard functionalities. On the other hand, the lecturer is provided with an illustrated graphic that gives a deeper insight into the functionality and further information about the adapter. Thereby, the lecturer is shown which possibilities an installation of this adapter opens up and how it can be used in the created course.

The implementation and presentation of the course overview from the lecturer's perspective are shown in Figure 4.7. Here, some information was mocked up during the implementation. By clicking on the individual adapters in the '*Create Content*' area, lecturers should be able to upload educational material and thus make it available to students. This functionality is mocked up in this prototype. On the right side of the overview, the lecturer gets some information about adapters installed in the course and additional adapters suggested by the system. In addition, statistics about students in the course are displayed in this course overview. These have been mocked up for demonstration purposes and are merely placeholders for other information.

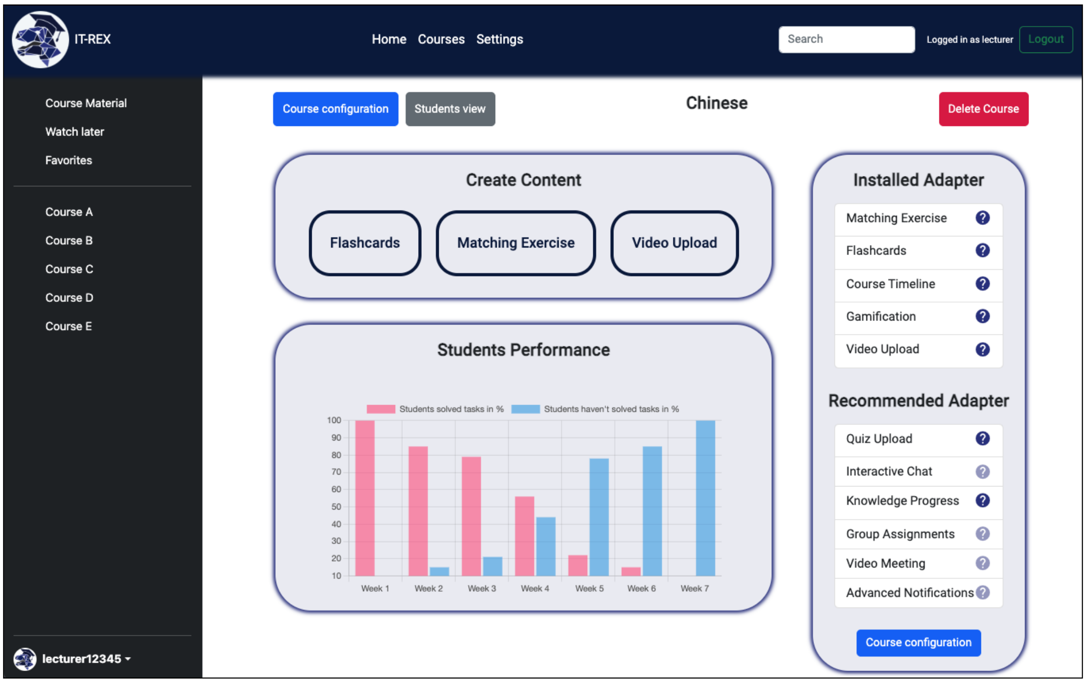

Figure 4.7: Implementation of the course overview functionality from lecturer's perspective.

Figure 4.8 shows the course configuration. In addition to the ability to view and edit course information, the installed and uninstalled adapters are also listed. With one click, the lecturer can add new adapters and thus functionalities in the course and remove the installed ones just as quickly. The specific configuration can be adjusted with one click on each adapter's configuration. The individual adapter configuration to which lecturers are redirected is shown in Figure 4.9.

The adapter configuration is shown in Figure 4.9, which depicts the configuration of the gamification adapter. On the right side, the correlating adapters are displayed so the lecturer can link adapters together with a single click or remove existing links accordingly. These links are not only established from one side but have been implemented so that the backend links the adapters from the other side. When opening the respective other adapter configurations, the adapters would thus also be linked. On the left side, the implementation shows the configuration options. These begin with the settings of the selected adapter. Below that, the frontend analyses the adapters linked to the adapter and thus offers options for configuring specific combined functionalities. Underneath this is the option for lecturers to present the adapter and its functionality respectively in the student's dashboard.

Figure 4.10 shows the implementation from the student's point of view. The courses created by the lecturer are displayed to the student in his overview. Other functions are mocked, such as joining other courses and the performance statistics displayed below. Thus, the user can only access the courses created before. By clicking on the course created by the lecturer, the student is directed to the course home page, including the dashboard, which is shown in Figure 4.11.

The dashboard and thus the course's home page for students is shown in Figure 4.11. The frontend implementation obtains information from the backend about which adapters the lecturer wants to display in the dashboard and presents them to the student accordingly. All content shown in the figure is mocked up and demonstrates possible ways of using such a dashboard.

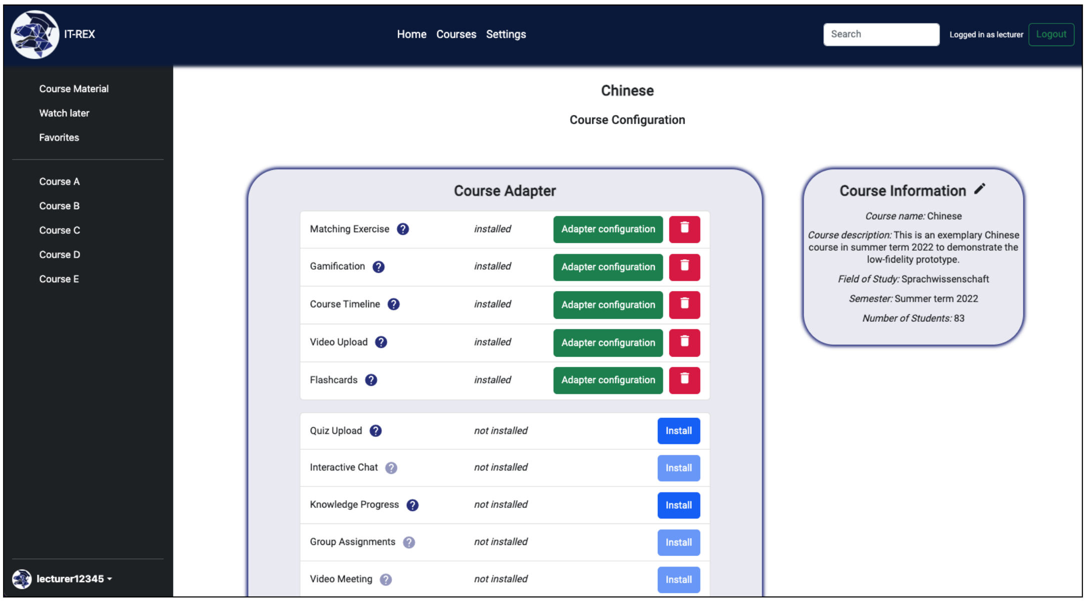

Figure 4.8: Implementation of course configuration functions.

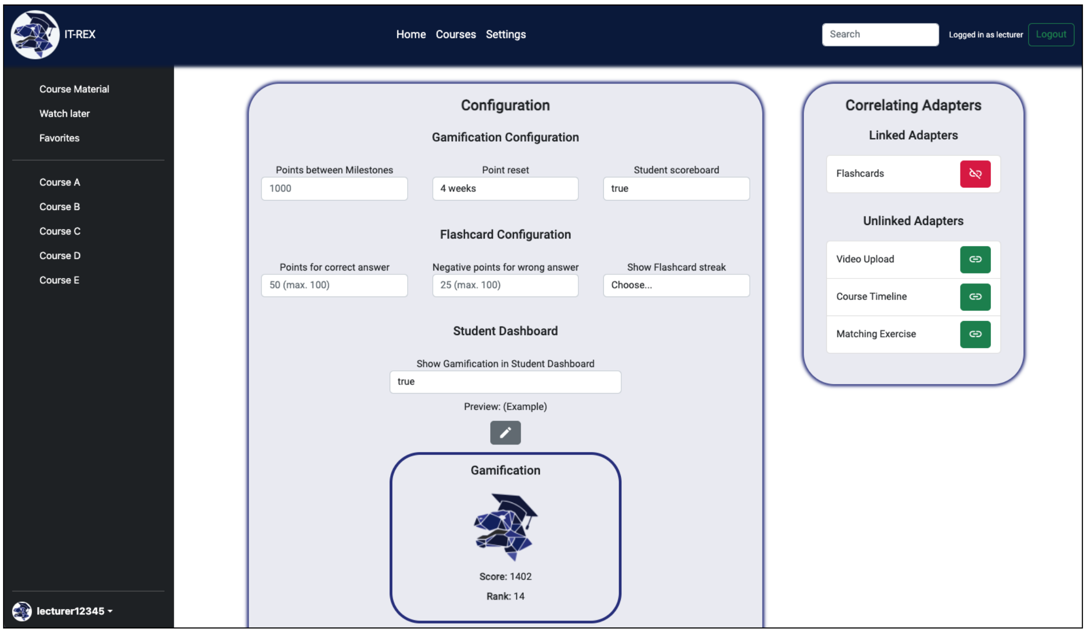

Figure 4.9: Implementation of the adapter configuration functionality.

The goal of the dashboard, as described in the concept, is the possibility for students to know exactly where to learn and improve on which topic. Thus, the dashboard not only offers the possibility to solve flashcards or to work on quizzes but also to display the learning status and integrate gamification elements, e.g., a student scoreboard. Quickstart elements are also integrated into the footer, which can be used to open the functions of the individual adapters.

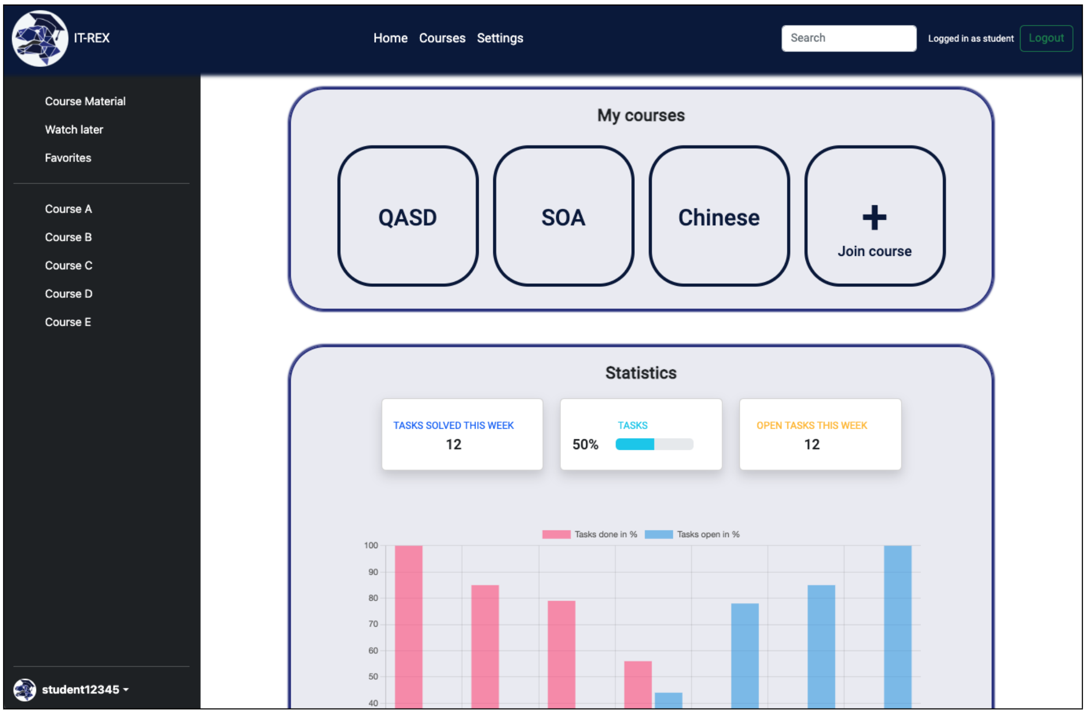

Figure 4.10: Implementation of the student course overview function.

The quickstart elements functionality is also mocked up in this implementation. For demonstration purposes of the prototype's features, the functionality of the '*Course Timeline*' adapter is implemented. The student is redirected to the course timeline by clicking on the corresponding tile, which is shown in Figure 4.12.

Figure 4.12 illustrates the functionality of the course timeline adapter and how it can be integrated into the platform. The timeline consists of weeks, which are connected with nodes. Per week, the lecturer can assign educational material and thus instruct the student on which tasks must be completed by when to stay within the course timeline. Here, the colors of the nodes correspond to the student's progress within a course. The frontend implementation captures the linked adapters and represents them as the week's assignments. All further information in the adapter is mocked up.

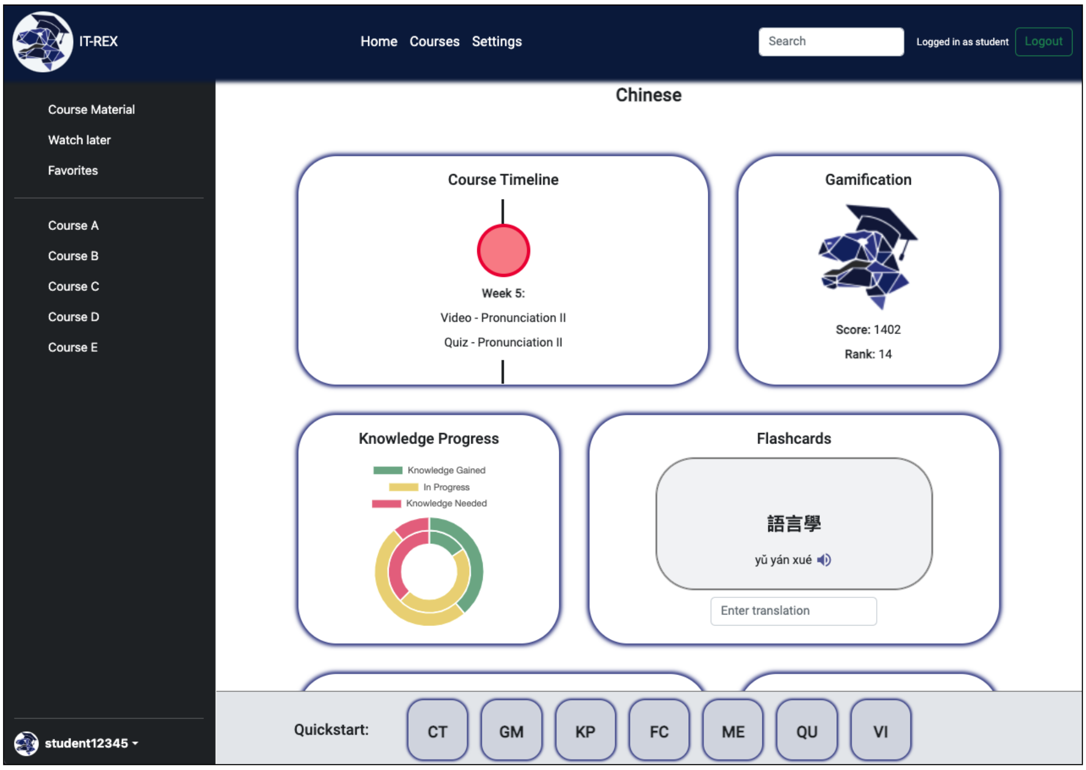

Figure 4.11: Implementation of dashboard functionality for students.

Figure 4.12: Implementation of the course timeline function.

4.3 Used Tools and Technologies

This section describes the tools and technologies used to develop the low-fidelity prototype. First, the choice of programming language for the backend is outlined. Next, the utilized libraries are described, and finally, the reasons for the applied frontend framework are explained.

For the backend, the microservice architecture outlined in the concept was chosen to make the prototype as close to reality as possible and to build functionally separated services. Each of these microservices was set up with the *Spring Boot Framework*[4] using the JAVA programming language. This decision had less of technological background. Instead, this constellation was chosen based on existing experience. In addition, implementing the backend with the *Spring Boot Framework* is a common and best practice, especially when dealing with microservices. Based on existing experience, *Apache Maven* was chosen as the build tool and a *MySQL*[5] database for an uncomplicated integration.

All microservices can be launched in *Docker*[6] and can be deployed in a *Kubernetes*[7] cluster with a few adjustments. The corresponding *Docker* files have already been created and are included in the source code of the individual microservices. Due to the timeframe of this thesis, the prototype was designed to be deployable in *Docker*. However, some adjustments are necessary, such as integrating a gateway to simplify the communication between the microservices and loosely couple them.

The libraries used in the backend extend the functions provided by *Spring Boot*. The database interface *Jakarta Persistence API (JPA)*[8] was used to simplify the transfer of data into and out of the database. For good handling and streamlining of the source code, *Lombok*[9] was used to simplify and shorten the source code with annotations. Furthermore, the library *Spring WebFlux*[10] was integrated for the use of web services with REST and communication with other microservices.

HTTP requests in the REST paradigm were used for communication between microservices and the frontend, as well as between the microservices themselves. For this purpose, the microservices provide an address and port for incoming HTTP requests.

The frontend of the prototype was developed in the TypeScript-based web applications framework *Angular*[11]. *Angular* was chosen as it has a large support community and is easy to set up and implement. The framework can be used to quickly generate the UI, which is a tremendous advantage for developing the low-fidelity prototype. In addition, the use of the MVC pattern in *Angular* is applicable due to the framework's organization of the source code. Hypertext Markup Language (HTML) and TypeScript are separate and can exchange information dynamically, allowing the controller to prepare and render content for the view in HTML. Another reason for using *Angular* is the already existing experience with the framework.

[4] https://spring.io/projects/spring-boot
[5] https://www.mysql.com/
[6] https://www.docker.com/
[7] https://kubernetes.io/
[8] https://spring.io/projects/spring-data-jpa
[9] https://projectlombok.org/
[10] https://spring.io/guides/gs/reactive-rest-service/
[11] https://angular.io/

To implement mock-ups and access different elements, *Angular Material Design*[12] was used in some parts of the frontend. In addition, some content from the Cascading Style Sheets (CSS) framework *Bootstrap*[13] was used to implement and mock up some elements of the frontend quickly. Finally, for the integration of the charts, which were inserted into the prototype for demonstration reasons, the library '*ng2-charts*'[14] was used, which integrates charts based on *Chart.js* in *Angular*.

[12] https://material.angular.io/
[13] https://getbootstrap.com/
[14] https://www.npmjs.com/package/ng2-charts/

5 Evaluation

This chapter describes the evaluation and assessment of this thesis. First, an overview of the structured evaluation process of this thesis is provided and described in Section 5.1. Then, Section 5.2 outlines a Goal Question Metric (GQM) for the evaluation. Section 5.3 focuses on the case study conducted with lecturers to evaluate the functionality and applicability of low-code adapters in the LMS. Here, feedback and results from lecturers and other stakeholders are presented and analyzed. All evaluation participants expressed valuable feature requests, which are also presented in this section. Section 5.4 validates and discusses the case study results. Finally, Section 5.5 identifies potential threats to validity.

5.1 Overview of the Evaluation Process

The evaluation of the e-learning platform concept with low-code adapters, including the low-fidelity prototype, was a structured process, which is shown in Figure 5.1. The input for the evaluation of the elaboration was formed by the two main components of this thesis, the concept itself and the low-fidelity prototype. Furthermore, since the prototype covers large parts of the concept, the conceptual background of individual functionalities was pointed out during the examination of the prototype.

The actual evaluation is divided into three parts. First, the GQM was conducted to check the objectives of the concept against the measurements and the evaluation. In this context, the GQM was also used to clearly define the structure and objectives of the evaluation, ensuring that the metrics described could be used to answer the questions posed and thus achieve the objectives. The GQM is discussed in more detail in Section 5.2. For the case study, three independent employees of the University of Stuttgart were chosen to perform a qualitative analysis of the concept and the prototype. The employees were chosen based on a wide range of experience and areas of specialization so that a broad spectrum of stakeholders was covered, and the results were valuable. The case study included a modified form of guided observation of the prototype so that participants could voice their opinions, comments, criticisms, and other thoughts aloud. This qualitative evaluation method evaluated aspects of usability, feasibility for lecturers, and possible use under realistic circumstances. After the prototype and concept presentation, the participants were asked to answer a questionnaire asynchronously after the meeting. This questionnaire was used to compare opinions and evaluations and to put them in relation to each other. The structure, the case study, the questionnaire, and the results and the evaluation are described in detail in Section 5.3. Furthermore, improvements and extensions mentioned by the participants, as well as other demands and requirements, are also addressed in Section 5.3.2 and the chapter about future work.

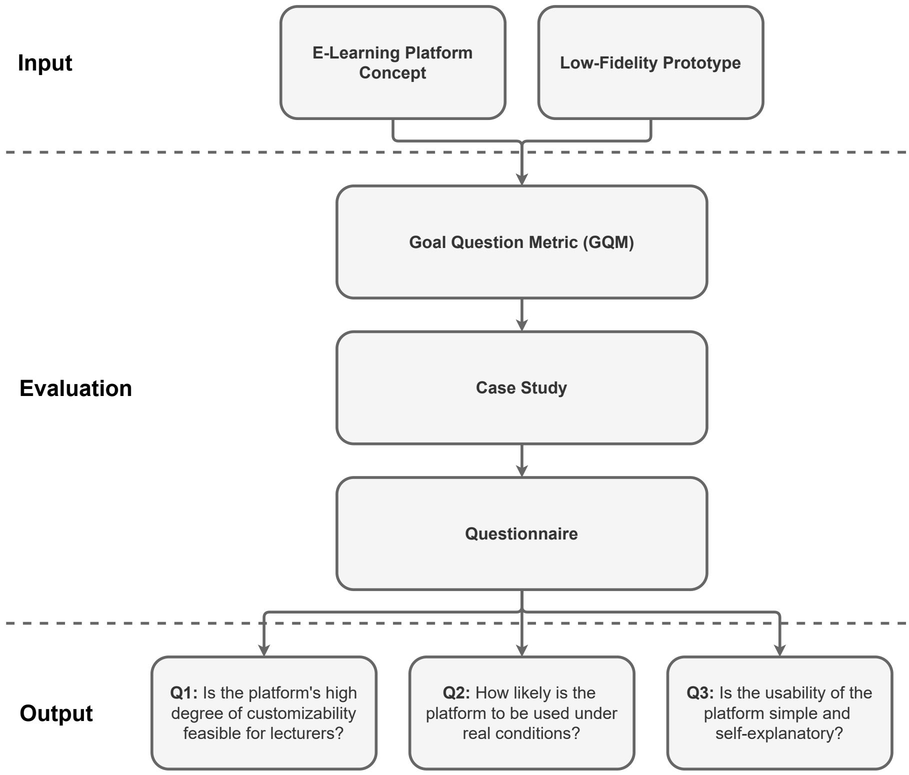

Figure 5.1: Evaluation process.

The output of the evaluation through the GQM, case study, and questionnaire can answer the following questions:

(Q1) Is the platform's high degree of customizability feasible for lecturers?

(Q2) How likely is the platform to be used under real conditions?

(Q3) Is the usability of the platform simple and self-explanatory?

5.2 Goal Question Metric (GQM)

A goal-oriented approach was chosen for the evaluation with the execution of a GQM. For this purpose, based on the requirements for the concept stated in this elaboration, goals were set up, which the concept had to achieve to be considered as approval of the concept. To achieve the PoC, it was also necessary to evaluate whether the concept developed in this elaboration solves the problem described initially, thus adding educational value. The process of GQM is illustrated in Figure 5.2 and includes goals, questions, and metrics. These defined goals emerged from the concept's stakeholder requirements. The questions were elaborated based on these goals, setting considerations on how to answer them through metrics.

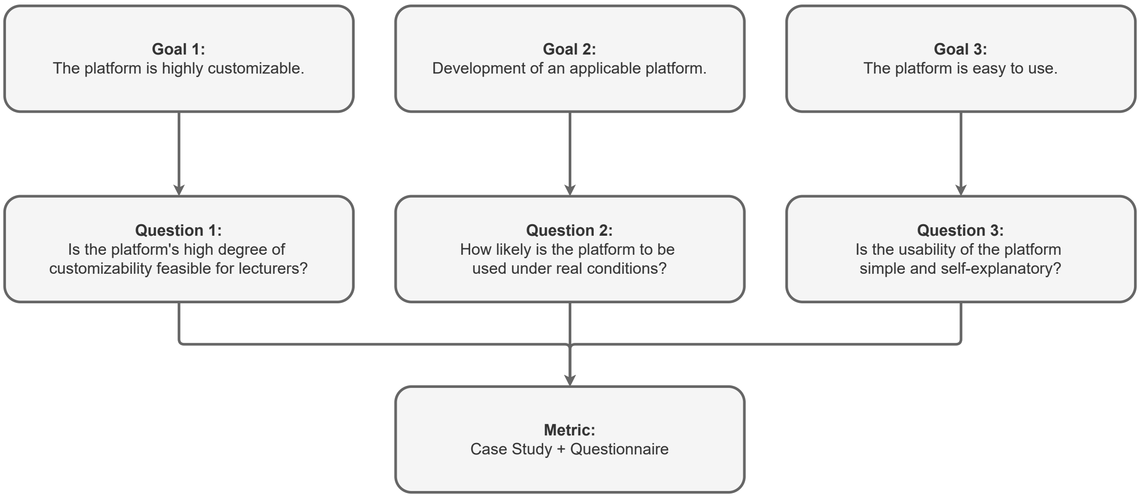

Figure 5.2: Goal Question Metric (GQM) for the evaluation.

The considerations resulting from the GQM outlined the following goals of the concept:

(G1) The platform is highly customizable.

(G2) Development of an applicable platform.

(G3) The platform is easy to use.

To achieve the defined goals, questions were set up at the operational level, which the evaluation must answer. These questions *Q1*, *Q2*, and *Q3* have already been presented in Section 5.1 and are also shown in Figure 5.2.

Metrics were established to evaluate the identified questions in the best possible way. These metrics include the execution of both a case study and a questionnaire. The type of case study was chosen to ensure that all participants could contribute their individual experiences and specialization areas as feedback and evaluation. Moreover, a questionnaire was required to answer the questions raised, which allowed a uniform evaluation of the views given by the participants. A detailed description of the case study and the questionnaire is explained in Section 5.3. This chapter also describes the feedback received, the analysis of the answers to the questions and the results, and to what extent the goals defined here could be achieved through the concept of this study.

5.3 Case Study

This section describes the case study of this thesis, which was conducted to evaluate and validate the e-learning platform with low-code adapters concept and its implementation in the low-fidelity prototype. First, Section 5.3.1 outlines the study design. Then, Section 5.3.2 describes the results with the evaluation findings and functional requirements. Finally, Section 5.3.3 relates the case study findings and discusses the results. The detailed process, including all questions asked to the participants in the case study, is published on *Zenodo*[1].

5.3.1 Study Design

The case study was conducted through guided interviews focused on usability and application in higher education. The e-learning platform with low-code adapters aims to be used domain-independently in a wide variety of study areas, and the participants in the case study were selected accordingly. For the evaluation, three employees of the University of Stuttgart were consulted who can evaluate the concept and the low-fidelity prototype from different perspectives. Since the evaluation was exclusively qualitative, a case study with three participants from different fields was possible. First, a lecturer from the field of linguistics was selected to cover a stakeholder from the non-IT field. The same participant was also involved in the initial requirements engineering. The second participant was from university didactics, advising lecturers on didactic measures and providing training and seminars. This participant had also previously participated and expressed requirements for such an e-learning platform. Finally, the third participant in the case study is a professor in teaching and learning with intelligent systems. Therefore, this participant assessed the concept from the perspective of the psychology of learning. The participant had no previous contact with this elaboration, thus allowing her to assess the concept and the prototype without prior knowledge.

In the interviews with the participants, the prototype described in Chapter 4 was examined. Several scenarios were carried out using the prototype, which illustrated the process of using the platform by lecturers, from the creation of new courses to the preparation of educational material and the view of the content from the perspective of the students. The prototype was presented to the participants without allowing them to use it independently. This decision was made since, in addition to the visual presentation of the aspects, the conceptual background of individual functionalities also had to be presented. The interviews were conducted partly via an online video conference and partly on-site at the University of Stuttgart.

The participants were presented with six scenarios in which they had to evaluate the aspects and functionality shown. In the process, the participants were given a rough overview of the functionalities shown but with as little information as possible so that their impression was minimally affected by the information provided. At the same time, the participants were encouraged to engage in dialog and freely express their thoughts during the presentation of the prototype or ask questions if anything was unclear. In addition, participants were asked prepared questions within the scenarios to obtain their views and opinion on certain aspects. These aspects are explained in detail in the subsequent sections.

[1] https://doi.org/10.5281/zenodo.7065141

The first scenario presented dealt with creating a new course and the first point of contact with the installable low-code adapters based on the required functionalities. Particular focus was placed on the adapter options and inquired whether these were integrated into the platform comprehensibly. In addition, part of the scenario was to receive adapter suggestions when the lecturer selects specific courses.

In the second scenario, the help that lecturers get from the platform to fully understand the low-code adapters was presented. For this purpose, the overview page of an adapter was shown, in which the functionalities of the adapter were presented, as well as possible combinations with other adapters. Again, the focus was on the level of detail and comprehensibility.

The following scenario describes the state of the course after its creation. The course is created, and the lecturer is on its home page, can view the installed adapters and create course material. This scenario examined whether the contents shown are necessary to simplify the lecturer's work.

The fourth scenario involves course modifications, so lecturers may want to extend or reduce the course's functionality and install or uninstall low-code adapters into or out of the platform. Again, greater focus was placed on the understandability of the ability to customize the course.

One of the most critical scenarios was the adapter configuration. This involves lecturers adjusting the parameters of the features and thus being able to modify the functionalities. This required looking at the comprehensibility of the configuration options and how the linking of adapters works within the platform.

The last scenario dealt with the e-learning platform from the student's point of view. Students use the dashboard in a course to work on educational material and to access the functionalities and information provided. The focus was on whether the dashboard was clear and easy to understand and whether the features were functional.

After each interview, the participants received a questionnaire, which they had to complete asynchronously immediately afterward. The questionnaire consisted of both closed and open questions. The closed questions were on a Likert-type scale ranging from '*strongly agree*' to '*strongly disagree*'. The questionnaire was divided into two sections. The first section of each participant's questionnaire consisted of 16 general questions about the concept, which were asked in the same form for each participant. The second section included specifically tailored questions. In this section, the linguistics lecturer was asked more specific questions about the direction of her courses and how she would use the platform in them. For the staff member in the area of university didactics, more questions were asked about how the platform supports the lecturers and whether this brings added value to the creation of educational material. For the professors in the area of teaching and learning with intelligent systems, questions were mainly asked about the direction of learning psychology and whether this platform has the potential to be beneficial for students.

5.3.2 Results

This section describes the results collected through the case study and the survey. The results are shown in the mind map in Figure 5.3. The section is divided into the individual areas of the LMS that were examined. The expert interviews gave comments, criticisms, and suggestions for further enhancements and adjustments. These are also presented in this section.

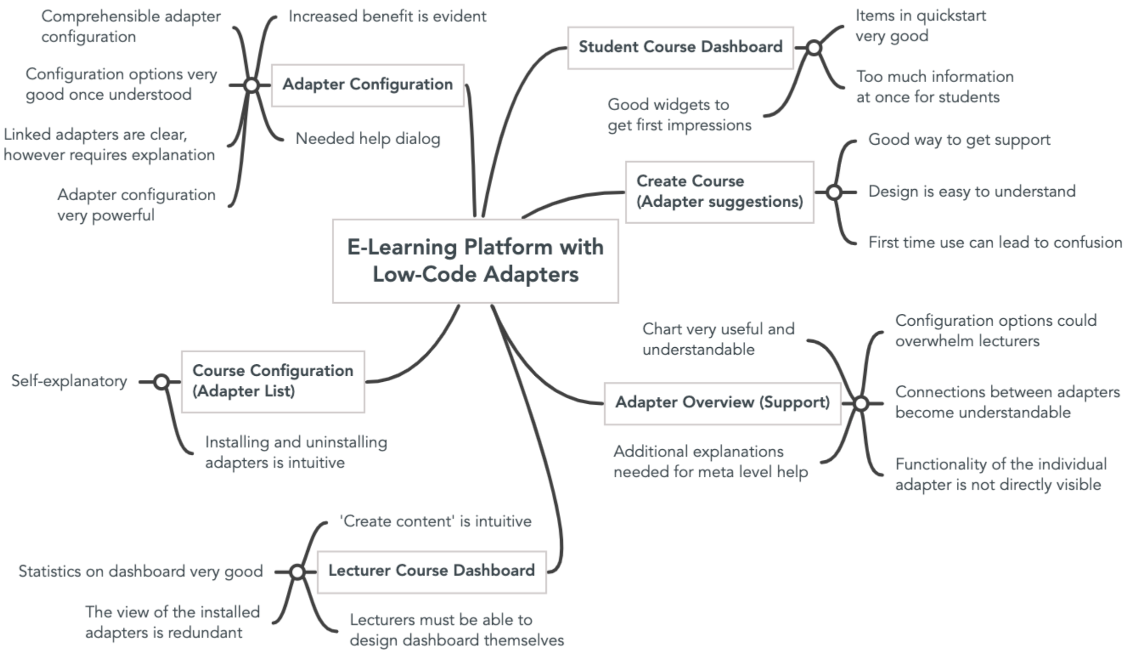

Figure 5.3: Mindmap of the participants' feedback.

The mind map in Figure 5.3 shows the different aspects of the concept and the investigation based on the prototype. Both positive and negative feedback from the experts were transferred to the figure. All participants' responses to the questionnaire can be reviewed on *Zenodo*[2].

Create Course (Adapter Suggestions)

The course creation feature, including the adapter suggestions, was rated very helpful and intuitive. However, a few points were noted where participants pointed out a possible lack of understanding. The adapter list, both the entire list and the suggested adapters, integrate very well on the course creation page and provide an advantage in setting the framework for the course right at the beginning. All participants also agreed that the design of the lists and usage is easy to understand, assuming the general integration of the adapters into the platform or a course is understood. Participants also considered the information about adapters on the course creation page sufficient. The available information is sufficient primarily when lecturers have worked with the platform before. On the other hand, it was criticized that the prototype's help and guidance were insufficient for a first-time user or a lecturer who is using the platform for the first time. The lecturers could be overwhelmed by it since they do not know which functionalities an adapter brings along and how it integrates into the platform.

Adapter Overview (Support)

The participants perceived the adapter overview as very positive after their review. Since the page acts as separate support and only provides information, lecturers are forced to visit the page when they need support.

[2] https://doi.org/10.5281/zenodo.7065141

The diagram was considered very helpful and comprehensible, clearly showing the interactions between the individual adapters. However, what information was exchanged between each adapter was not directly visible. On the other hand, it was criticized that the presentation could lead to an overload of the lecturers, as so much information is provided simultaneously. However, this is predominantly due to the chosen UI and could be remedied by making it more concise or appealing. This also applies to the functionality of the individual adapter and the configuration options, as lecturers must first familiarize themselves with the layout. It was also brought up that an explanation of adapter interactions would make sense on a meta-level and could be placed on this Adapter Overview page.

Lecturer Course Dashboard

Regarding the lecturer's course overview, positive and negative statements were made about the content and the presentation. The layout of the create content area was appreciated and described as very intuitive, as it makes it easier for the lecturer to use. Evidently, the lecturer could create content independently, whether by uploading videos or quizzes or by creating flashcards. In addition, the section for students and course statistics was found to be extremely helpful, as lecturers would like to see an analytics tool in e-learning platforms to collect and display meta-information about the use of learning opportunities. Moreover, the demand was expressed to be able to design the lecturer dashboard individually and to display functions on it according to their interests. Furthermore, there was a request for the ability to adjust the course timeline or modify individual adapters directly on the dashboard page without necessarily having to create or add new content. On the other hand, the area with the installed adapters and the additional adapter suggestions was rated as unnecessary and redundant. All participants pointed out that the adapter overview within a course is sufficient in the respective preferences and that the space in the dashboard could be used otherwise. However, the comment was brought up that a list of installed adapters may be helpful if a lecturer has not accessed a course for an extended time and thus is reminded of what functionality the course offers.

Course Configuration (Adapter List)

The functionality of the course configuration, including the list of installed and uninstalled adapters, only received positive feedback. The design is self-explanatory, and installing and uninstalling adapters in a course is straightforward. The simplicity of the one-click installation was highly appreciated.

Adapter Configuration

The participants' comments on the adapter configuration were largely positive. It was mentioned that the advantage of the configuration options of individual adapters for combinable adapters is evident and comprehensible in principle. Nevertheless, it was expressed that adapters with multiple connection options offer so many configuration options that the UI could be overloaded. Too many configuration options could lead to a lack of understanding. Fundamentally, the functionality and the idea of the configuration options of different features and linked adapters were considered very powerful and, for the most part, easy to understand. The linking of individual adapters was also clearly illustrated, but the participants expressed a need for an initial explanation and support to understand the adapter links. However, on the other hand, it was criticized that lecturers need to be supported with information during linking and configuration, for instance, with a help dialog.

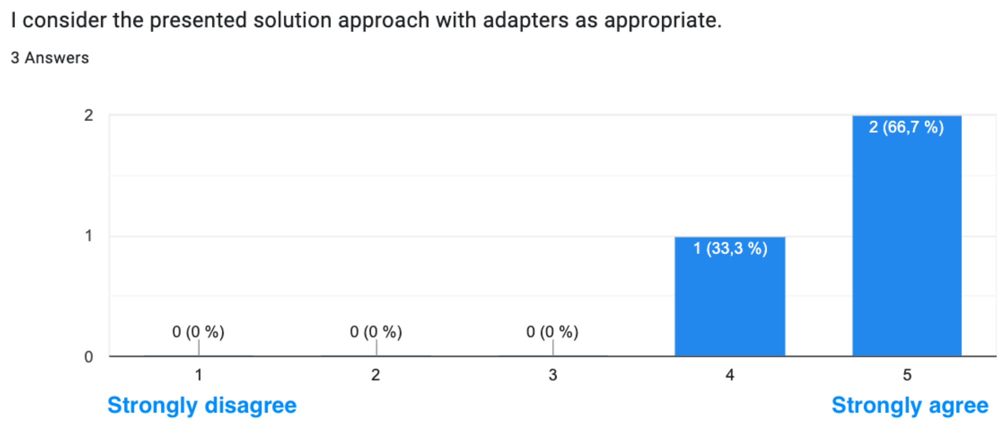

Figure 5.4: Questionnaire result for the appropriateness of the adapter approach.

In addition, further explanations are required to make it clear to the lecturer which adjustments are being made to which configuration option. Especially when using the various configuration possibilities for the first time, lecturers who are not familiar with IT might have significant difficulties.

Student Course Dashboard

All participants agreed that a dashboard would benefit the students a lot. The ability to show students the current topics on a dashboard and motivate them to learn is the best way a course should be presented in an e-learning platform. The quickstart area shown in the prototype was perceived as very good, as functionalities and teaching offers can be accessed quickly without having to search for the corresponding adapter beforehand. The widgets on the home page with the educational material and the option of solving flashcards and other tasks were also widely perceived as very positive. On the other hand, it was criticized that students could be overwhelmed by the amount of information and possibilities and a large number of widgets. So, there should be limited information displayed.

Usability

The usability of the entire prototype was perceived as very positive and largely self-explanatory. However, some points were mentioned that were considered very well implemented, others that need improvement. All participants of the evaluation appreciated the UI and expressed that a realistic picture of a future implementation in the LMS was imaginable. On the other hand, some improvements and issues regarding the UI have been mentioned, especially in the color or shape of the design.

In the questionnaire to the participants, several questions were asked about usability, which provide an overview regarding the use of the concept and the platform. Figure 5.4 shows the questionnaire results, how the participants assess the use of adapters to utilize functionalities, and how well they were implemented. The results indicate that adapters are appropriate for this type of application in a LMS.

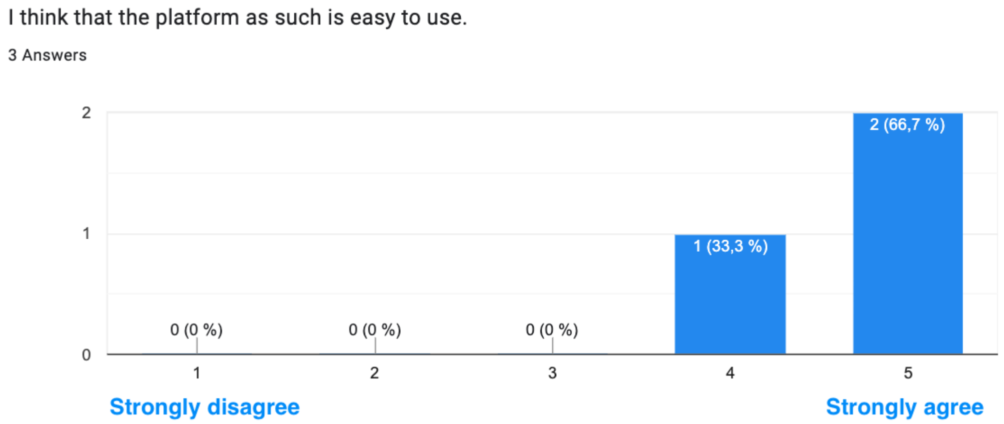

Figure 5.5: Questionnaire result for the easiness of use.

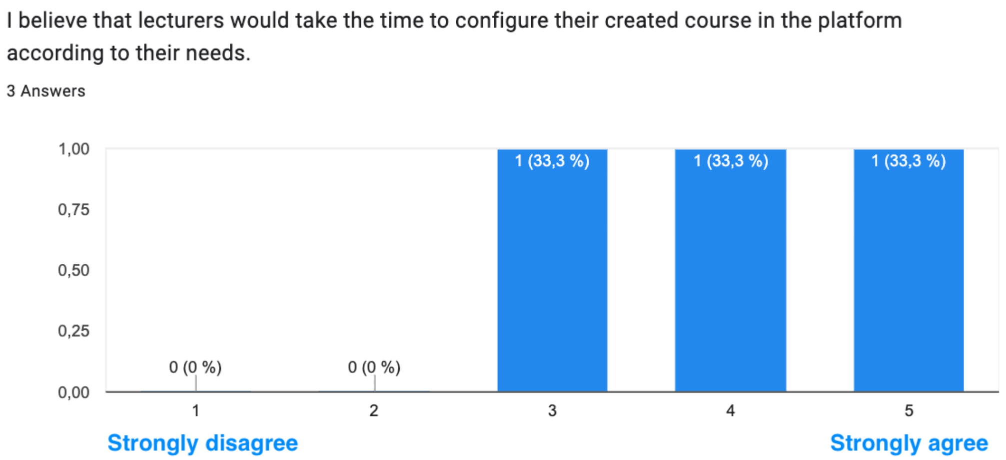

Figure 5.6: Questionnaire result for the patience of the lecturers who set up their course.

Figure 5.5 confirms the statements made in the case study about ease of use. The diagram shows that two participants strongly agree that the designed platform is easy to use.

Despite the unanimously positive results in terms of the usability of the system, the opinions on real-world usage and applicability differ for each participant. For example, Figure 5.6 shows the participants' assessment of whether lecturers would take the time and have the patience to configure the platform to their requirements. The question does not consider whether lecturers have possible assistants or receive other support in creating courses. Results ranged from neutral to strongly agreeing that lecturers would take the time and make an effort to design their courses.

The findings, shown in Figure 5.7, also reflect the results of the case study. The results were very different regarding whether the process for lecturers, from creating the course to uploading educational material, is comprehensible. One participant voted against it, while the other two tended more toward being more understandable.

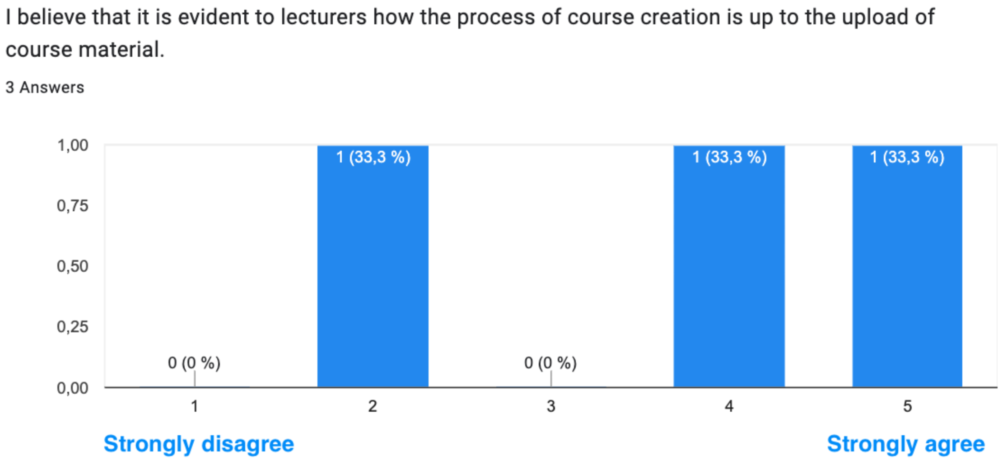

Figure 5.7: Questionnaire result for the process of using the platform.

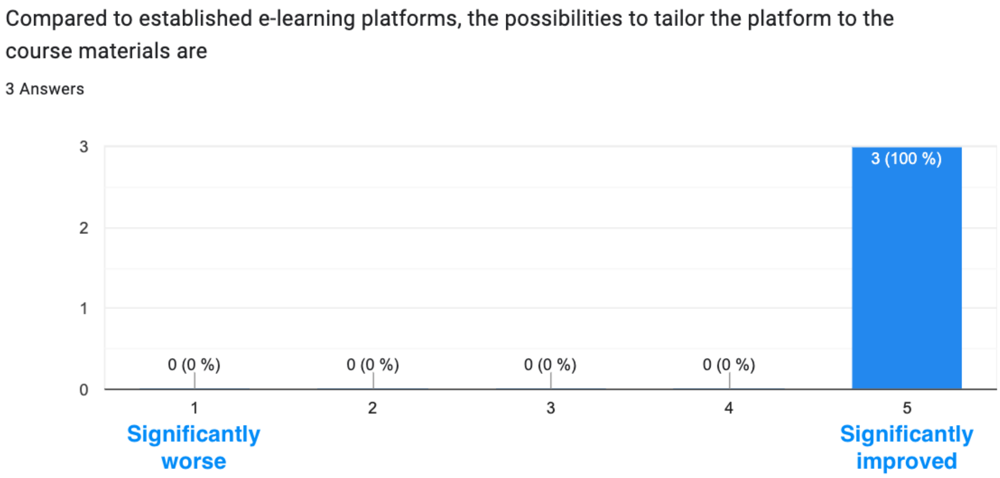

Figure 5.8: Questionnaire result for improvement compared to established platforms.

Figure 5.8 compares the concept of this e-learning platform with low-code adapters to established LMSs. The question focused on whether the platform can be tailored to educational materials better than established platforms. The result was unambiguous, and all participants stated that the platform developed in this thesis significantly improves the possibilities of adaptability.

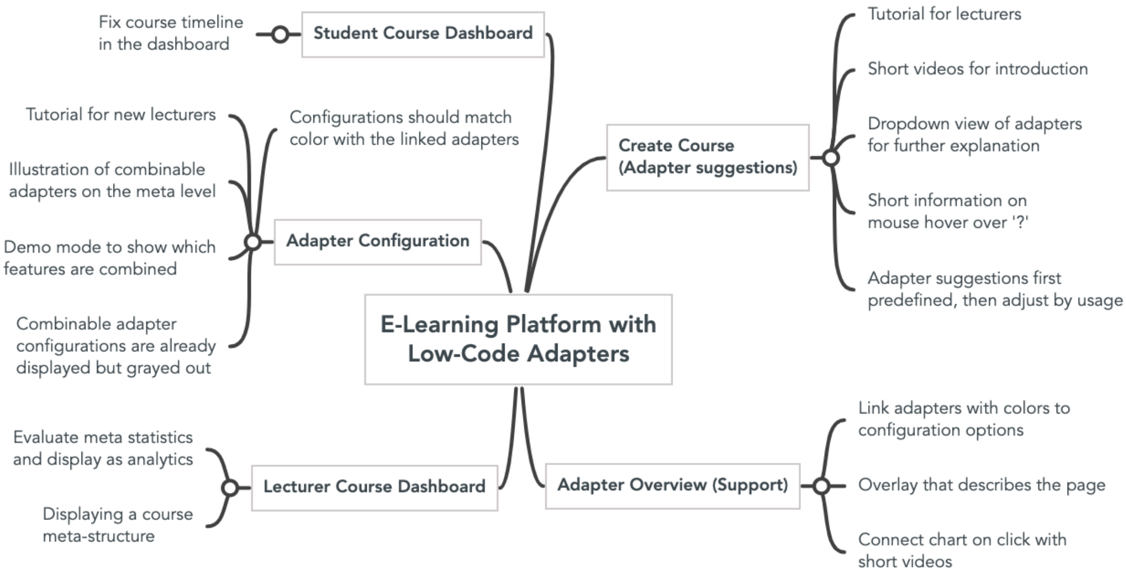

Figure 5.9: Mindmap of the participants' feature requests.

5.3.3 Discussion

In this section, results from the case study and the associated questionnaire from Section 5.3.2 are discussed, compared, and evaluated. The validation of the hypotheses is discussed and examined in the following Section 5.4.

The case study was a success, as the results show a clear tendency that an e-learning platform would benefit from this concept and be used in higher education. In addition, there is a lot of positive and negative criticism of features and implementations within the platform, so the evaluation of this elaboration also makes numerous suggestions for improvement. On the one hand, it was shown which aspects of the concept have already been implemented well, and on the other hand, which areas need to be improved to offer lecturers and students a modern, attractive learning platform.

Aspects regarding the platform's usability are important outcomes for further development and elaboration of the implementation. Nevertheless, aspects such as the UI design and chosen colors are to be considered that not too much importance was placed on a visually appealing UI in the low-fidelity prototype. In addition, some criticisms that were voiced were less related to the concept as such but rather to the content of an e-learning platform that did not originate with the concept and was only used for demonstration purposes in this elaboration.

Figure 5.9 includes some feature requests from participants for each of the areas presented in the case study. The statements made in the case study about the course creation feature with adapter suggestions are good, but the participants expected more explanations and information for lecturers. From this, the need was derived that there should be a tutorial for lecturers, either in the platform with an overlay or as short videos to explain the platform. There was also a demand to have short descriptions of the adapters already displayed on the course creation page, either as a dropdown under each adapter or on a mouse hover over the '?' (help area). The suggestion also came from the participants to initially pre-set the suggestions for the adapters and then adjust them autonomously via the analysis of the usage.

Some suggestions for improvement were made for the adapter overview, i.e., the adapter support page, which would integrate well into the existing platform. The assessment of the participants is valid since the more adapters can be linked together, and the more configuration options, the more confusing the overview page becomes. In addition, some participants brought up valuable feature requests that could fix this problem. For one, linkable adapters could be highlighted in color in the diagram, and the associated configuration options with the same color. Further, a short video could be played when lecturers clicked on a link between adapters in the diagram. Another option would be an overlay for lecturers that describes the individual areas of the page and explains to the lecturers where they can see the information they need.

The feedback on the lecturer's dashboard was consistently positive, but further developments were suggested. For example, the participants requested an analysis of meta-statistics about the course and the students on the lecturer's dashboard page. The student statistics were only used in the prototype as a placeholder for further developments. There was also a demand to identify where lecturers could access individual adapters directly.

It was also suggested that linkable adapters be displayed on a meta-level to show the link and the information being conveyed clearly. A supporting feature was voiced to make the process easier for lecturers. Within the configuration options of an adapter, the options should be grayed out when there is no link and activated when adapters are linked together.

As mentioned earlier, participants pointed out that a student dashboard could become overloaded. This statement is valid depending on the number of adapters installed and the use of the dashboard widgets. One participant expressed a demand to be able to set the course timeline, and thus any other adapter, as a static home page for students so that they would only see the information for that one adapter.

5.4 Validation

In this chapter, the concept is validated using the case study and the questionnaire to test the hypotheses of this elaboration. The validation is based on the contents of the GQM, described in Section 5.2. Based on the three questions in the GQM, the following hypotheses were formulated:

(H1) The platform's high degree of customizability is feasible for lecturers.

(H2) The platform is applicable in higher education under real conditions.

(H3) The platform is self-explanatory and easy to use.

The goal of validation is to either accept or reject the hypotheses related to the case study. The hypotheses are partially interrelated, such as usability by lecturers and applicability in higher education, so careful consideration must be given to how a rejected hypothesis affects the others.

As described in Section 5.3.2, the platform was highly beneficial for lecturers to teach, and the platform's customizability could be ideally used for educational material. It was discovered through the case study that lecturers want to adapt their platform so that it is tailored to the content to be beneficial. It emerged that the platform is ahead of established LMSs in terms of customization. An argument against the first hypothesis is comprehension since, depending on lecturers, the high degree of customizability can lead to misunderstandings or problems when using the platform.

Nevertheless, the advantages of customizability outweigh the disadvantages. Since the case study was carried out with three experts, it must be mentioned that the evaluation only shows a tendency and does not reveal a statistically significant result. Nevertheless, the first hypothesis *(H1)* can be accepted after evaluating the case study and the questionnaire.

Regarding the second hypothesis *(H2)*, a statement can be made based on the circle of participants. The participants of the evaluation are experts in higher education and have, therefore, evaluated the concept and the prototype under the aspects of higher education. The experts considered the applicability and use under real conditions to be feasible. This is also reflected in the results from Section 5.3.2. The extensions and adaptations discussed in Section 5.3.3 should be implemented before the platform is used in real circumstances in higher education. However, the platform as such is already applicable and can be used. Thus, the second hypothesis *(H2)* is considered as accepted. However, this is also only a tendency and not statistically significant.

In terms of self-explanation and ease of use, and thus the third hypothesis *(H3)*, it can be stated that the ease of use was considered positive in the case study and the questionnaire. Some features of the prototype were criticized by the experts, which emphasized poor self-evidence. Nevertheless, the criticism was explicitly related to the low-fidelity prototype and less to the concept itself, so the criticism was expressed at a qualitatively very high level. Figure 5.5 also shows the overall opinion of the ease of use. So the third hypothesis *(H3)* is considered as accepted. This shows only a tendency, however, and is not statistically significant.

5.5 Threats to Validity

The following sections explain the potential threats to the validation of the study and the countermeasures taken. First, internal threats are listed in Section 5.5.1. This is followed by external threats in Section 5.5.2 and finally construct validity in Section 5.5.3.

5.5.1 Internal Validity

Even if the first hypothesis benefits from the validation of the second and third hypotheses, the applicability of the platform in higher education and an easy-to-use platform only influences the platform's feasibility by lecturers to a limited extent. Therefore, *(H1)* is partially dependent on the other hypotheses but may also be accepted if *(H2)* and *(H3)* are not accepted.

The explanations and descriptions of each scenario and content for the interviews and the prototype presentation took place under the same conditions for all participants. Nonetheless, some questions, statements, and viewpoints that arose in the first interviews were considered in the subsequent ones. These participants were confronted with already collected statements and were asked to present their points of view. Even if not intended, the questions then could be interpreted as suggestive. In addition, the interviews were not conducted in succession. This time delay might have led to minor deviations in the questions and the flow of the discussions.

5.5.2 External Validity

The main external threat is the small number of participants that have been interviewed. With three experts, the sample size is relatively small, even though these experts are from different areas of expertise. Therefore, the results of this thesis should be considered under the aspect that they only show an indication or a tendency rather than proven evidence. However, the sample size consists of experts from different areas, so various perspectives were presented in the evaluation. Nevertheless, it would be an appropriate countermeasure to interview more than one expert from the respective area in the evaluation, especially more lecturers, as they are the primary users of the platform.

Another external threat is that most of the participants had already been involved in the development of the concept and were, therefore, partly familiar with it. Thus, the participants were not independent employees of the university. To countermeasure this threat, a lecturer without prior knowledge of the concept and the prototype was also interviewed.

5.5.3 Construct Validity

The low-fidelity prototype was fundamentally implemented from scratch and did not include all functionalities of an e-learning platform due to the timeframe of the thesis. The focus was on integrating the low-code adapters, and many other functionalities and UI elements were mocked up. In addition, some content was adopted from IT-REX so that the prototype alone represents a PoC and is only suitable for evaluating the functionalities examined in this elaboration. Additional functionalities would have to be implemented in the platform for a more in-depth evaluation.

Second, the prototype was presented to the participants using a showcase, and they were not allowed to click through the prototype independently. Although participants were asked to think aloud, some prepared questions may have been phrased slightly different so that they appeared like closed questions. To counteract this, important questions were formulated before each scenario to keep them as open as possible.

In the interviews, it was difficult to obtain statistical data that could be evaluated and compared. The participants were encouraged to express their first impressions. For this reason, participants were asked to fill out a questionnaire asynchronously after the interview.

6 Conclusion and Future Work

This chapter addresses this thesis's essential aspects and findings and provides an outlook for future work. First, Section 6.1 presents the results and conclusions of this thesis and explains how these results can be used. Then, Section 6.2 lays out the groundwork for future work. This includes the next steps of the e-learning platform with low-code adapters, as well as further research on its use in higher education. Possible further developments of the concept and in-depth investigations of individual components are also part of future work.

6.1 Results and Conclusion

This section concludes the key aspects and insights of this elaboration. First, it briefly recaps the problem statement solved in this thesis. Then, the solution approach of the conceptual design is presented. Finally, the benefits of the concept are shown, and the results are summarized.

Established LMSs in higher education allows lecturers to provide educational material digitally to their students. The platforms usually have many functionalities and can be extended with plugins. These LMSs are pretty inflexible, and functionalities must either be used as provided by the platform or not used at all. Therefore, lecturers must adapt their educational material to the functionalities offered by the e-learning platform. Lecturers can add plugins as required, which are developed by third-parties and are neither customizable and configurable for the lecturer nor combinable and interactive with other plugins or other functionalities. These plugins are complicated and challenging to use and require more profound know-how. This makes it impractical for lecturers to use and, simultaneously, means unnecessary overhead.

The thesis aims to ensure that lecturers do not have to adapt their educational material to the LMS but rather adapt the platform according to their requirements. For this purpose, the solution approach was focused on an e-learning platform, which is equipped with fundamental functionalities. Moreover, the lecturers can add functionalities to the course according to their demands. An approach with LCDP is considered appropriate since most lecturers do not have any IT experience and thus cannot do any programming themselves. Lecturers should be empowered to compose and customize features with appealing UI elements. The platform's functionalities can be extended and are thus divided into low-code adapters. These adapters can be installed separately in a course but can be linked by the lecturers so that they can communicate with each other independently and provide combined functionalities. For a modern LMS, the gamification aspect is also taken into account, and content from the IT-REX project is considered. To validate this approach, a low-fidelity prototype was created. However, due to the timeframe of this thesis, the concept's main aspects were implemented, but some additional functionalities were mocked up. Thus, the thesis and the prototype provide an overview of the main aspects of the concept. Subsequently, the concept was evaluated based on a GQM and a case study with experts and lecturers.

The results obtained from the evaluation of the solution approach were very positive. The experts and lecturers interviewed in the case study were convinced by the concept and stated that tailoring a LMS to the requirements of the lecturers would be an essential step toward efficient e-learning. Despite the consistently positive feedback from the participants, it was pointed out that lecturers with no IT experience need to receive a different introduction and explanation of the platform than lecturers with IT experience. Feedback was also brought in that lecturers need to get more familiar with the platform when using it for the first time. They need to retrain themselves to add functionalities to the platform and customize it to their needs, rather than receiving standard functionalities from established LMSs that only meet their demands to a limited extent. Some participants suggested that lecturers must take the time or receive appropriate support to make this transition.

During the case study to evaluate the concept, the low-fidelity prototype and the concept were presented to the stakeholders so that they could express their opinions. After the initial discussions in the context of the requirements engineering process, the interviews during the case study clearly showed how different stakeholders' requirements for an e-learning platform could be. It is not only the lecturers and students who have requirements for a modern e-learning platform but also experts from different areas who try to improve higher education and have precise ideas of an optimal LMS. Some of these requirements can be well combined and integrated into the platform, while others collide with each other, making it challenging to combine them. For instance, on the one hand, there is a requirement for students to be able to customize their dashboard on the home page independently. On the other hand, other stakeholders prefer a student dashboard composed exclusively by the lecturer.

The concept elaborated in this thesis proposes a solution approach to equip an e-learning platform with reusable low-code adapters to simplify the handling and configuration options of the platform for lecturers. The concept is based on already existing LMSs but chooses an interdisciplinary composition of functionalities and content approach. A SPL is created, where each course gets a base platform and, in addition, a specific set of functionalities, depending on the requirements of the lecturers. This benefits both lecturers in course creation and tailoring to the course content and the students. Students receive a platform tailored to their performance that adapts to their learning as they progress through a course. Experts from areas of higher education didactics, vocational and technical pedagogy, and teaching psychology contributed to the design and evaluation of the concept. This allowed a high focus on achieving the best possible student and lecturer performance. The non-functional properties increase the platform quality, which benefits all stakeholders of the e-learning platform.

6.2 Future Work

The future work consists of different components, which show future investigations. This section describes the theoretical and practical continuation of the concept and the challenges that need to be solved.

First, the case study should be conducted with more than three participants so that the evaluation not only indicates a trend but also makes it possible to make statistically significant statements about its usability and applicability in higher education. Additional experts from the studied areas would be needed to analyze differences in expert opinions. An evaluation based on many lecturers from various fields of study would also be very beneficial. Through a large-scale study, indications could be gained as to which areas of the platform can be used and at which points the lecturers would have difficulties in using the platform.

The concept can be extended in many directions. In the current status of the thesis, the composition of the LMS from a base e-learning platform and low-code adapters was investigated. In addition, research can be done on how the base platform could provide external adapters or functionalities to students. The interactions between the platform and third-party applications can be investigated so that in a future state, for instance, a lab can be simulated for chemistry classes with the help of the platform, respective low-code adapters, and Virtual Reality (VR) or, at best, Augmented Reality (AR).

A possible further investigation could focus on handling a large number of adapters. The more low-code adapters are added to a course and linked, the potentially more confusing the platform can become for lecturers. This will require an investigation into the extent to which lecturers could handle a large amount of information and possibilities and how these could best be presented in the UI.

After the evaluation by lecturers and experts of different areas, an evaluation from the opposite side, the students, would be advantageous. The state of the concept in this elaboration is currently designed mainly for lecturers, as they can customize and tailor their courses. Nevertheless, lecturers can design the student dashboard, and the students can interact with it directly on their home page. From this perspective, an evaluation from students would be of great value.

In the future, the approach of the LCDP should be further enhanced, and the composition of a course should also be visually representable and adjustable via drag and drop. For this purpose, research must be conducted to evaluate the potential possibilities. In addition, a diagram is needed in which lecturers can visually see the currently installed adapters and adjust them within the diagram, i.e., add and remove adapters, as well as link and unlink adapters visually.

Based on the evaluation discussions with experts, some new demands were raised. One of these requirements was that the student dashboard should be permanently equipped with a course timeline feature to track assignments week by week. This functionality needs to be explored from the student side and involves lecturers' intuitive use and setting.

Lastly, the support options for lecturers should be investigated more in-depth. The feedback from the experts on the prototype was also that the lecturers would need a tutorial for the platform. This could be created with short videos, a demo mode, or an introduction within the platform. For this, the individual possibilities must be evaluated, and an assistance concept must be created.

Bibliography

[AA15] V. Arkorful, N. Abaidoo. “The role of e-learning, advantages and disadvantages of its adoption in higher education”. In: *International Journal of Instructional Technology and Distance Learning* 12.1 (2015), pp. 29–42 (cit. on p. 1).

[AE18] J. Y. Ahn, A. Edwin. “An e-Learning Model for Teaching Mathematics on an Open Source Learning Platform”. In: *International Review of Research in Open and Distributed Learning* 19.5 (2018). DOI: https://doi.org/10.19173/irrodl.v19i5.3733. URL: https://www.erudit.org/en/journals/irrodl/1900-v1-n1-irrodl04235/1055557ar/abstract (cit. on p. 12).

[AGB17] S. Azouzi, S. A. Ghannouchi, Z. Brahmi. “Software Product Line to Express Variability in E-Learning Process”. In: *Information Systems*. Cham: Springer International Publishing, 2017, pp. 173–185. ISBN: 978-3-319-65930-5 (cit. on p. 9).

[BBZ04] H. Balzert, H. Balzert, O. Zwintzscher. “Die E-Learning-Plattform W3L”. In: (2004), pp. 129–138. DOI: https://doi.org/10.1007/BF03250932 (cit. on p. 8).

[BHH20] S. Bai, K. F. Hew, B. Huang. “Does gamification improve student learning outcome? Evidence from a meta-analysis and synthesis of qualitative data in educational contexts”. In: vol. 30. Elsevier, 2020, p. 100322 (cit. on p. 10).

[BN20] K. Bernecker, M. Ninaus. “No Pain, no Gain? Investigating motivational mechanisms of game elements in cognitive tasks”. In: *Computers in Human Behavior* 114 (Aug. 2020), p. 106542. DOI: 10.1016/j.chb.2020.106542 (cit. on p. 10).

[BRH20] T. Beranic, P. Rek, M. Heričko. *Adoption and Usability of Low-Code/No-Code Development Tools*. English. 2020. URL: https://www.proquest.com/conference-papers-proceedings/adoption-usability-low-code-no-development-tools/docview/2531366275/se-2 (cit. on p. 9).

[CN13] S. Chimalakonda, K. V. Nori. “What makes it hard to apply software product lines to educational technologies?” In: *2013 4th International Workshop on Product LinE Approaches in Software Engineering (PLEASE)*. 2013, pp. 17–20. DOI: 10.1109/PLEASE.2013.6608657 (cit. on p. 10).

[Coo13] W. Cook. “Five Reasons You Can’t Ignore Gamification”. In: *Chief Learning Officer* 12.5 (2013), pp. 46–55 (cit. on p. 10).

[Dev97] P. C. Dev. “Intrinsic Motivation and Academic Achievement: What Does Their Relationship Imply for the Classroom Teacher?” In: *Remedial and special education* 18.1 (1997), pp. 12–19 (cit. on p. 10).

[Don14] P. Donohoe. *Introduction to Software Product Lines*. 2014. URL: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=423718 (cit. on p. 9).

[FKY18] M. Fırat, H. Kılınç, T. V. Yüzer. "Level of intrinsic motivation of distance education students in e-learning environments". In: *Journal of Computer Assisted Learning* 34.1 (2018), pp. 63–70 (cit. on p. 10).

[IEC11] IEC. "ISO / IEC 25010 : 2011 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models". In: 2011 (cit. on p. 19).

[Kap22] S. Kappagantula. *Everything You Need To Know About Microservices Design Patterns*. Mar. 26, 2022. URL: https://www.edureka.co/blog/microservices-design-patterns (visited on 09/30/2022) (cit. on pp. 26, 27).

[KAY14] G. Kiryakova, N. Angelova, L. Yordanova. "Gamification in education". In: Proceedings of 9th International Balkan Education and Science Conference. 2014 (cit. on pp. 3, 10).

[KS18] S. Krusche, A. Seitz. "ArTEMiS: An automatic assessment management system for interactive learning". In: *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. 2018, pp. 284–289 (cit. on pp. 4, 18, 26).

[MBH+20] M. Meeter, T. Bele, C. Hartogh, T. Bakker, R. de Vries, S. Plak. "College students' motivation and study results after COVID-19 stay-at-home orders". In: Oct. 2020. DOI: 10.31234/osf.io/kn6v9 (cit. on pp. 1, 3).

[Mun11] C. I. Muntean. "Raising engagement in e-learning through gamification". In: *Proc. 6th international conference on virtual learning ICVL*. Vol. 1. 2011, pp. 323–329 (cit. on p. 10).

[PBSG19] C. Ploder, R. Bernsteiner, S. Schlögl, C. Gschliesser. "The Future Use of Low-Code/NoCode Platforms by Knowledge Workers – An Acceptance Study". In: *Knowledge Management in Organizations*. Cham: Springer International Publishing, 2019, pp. 445–454. ISBN: 978-3-030-21451-7 (cit. on p. 9).

[Pio10] M. Piotrowski. "What is an e-learning platform?" In: *Learning management system technologies and software solutions for online teaching: Tools and applications*. IGI Global, 2010, pp. 20–36 (cit. on p. 8).

[PSR10] D. Pandey, U. Suman, A. Ramani. "An Effective Requirement Engineering Process Model for Software Development and Requirements Management". In: *2010 International Conference on Advances in Recent Technologies in Communication and Computing*. 2010, pp. 287–291. DOI: 10.1109/ARTCom.2010.24 (cit. on p. 13).

[PTP19] I. Petroulis, M. Tzelepi, K. Papanikolaou. "On the Design of Gamification Elements in Moodle Courses". In: *Games and Learning Alliance*. Cham: Springer International Publishing, 2019, pp. 428–437 (cit. on p. 3).

[RGGB19] J. Rodas-Silva, J. A. Galindo, J. García-Gutiérrez, D. Benavides. "Selection of Software Product Line Implementation Components Using Recommender Systems: An Application to Wordpress". In: *IEEE Access* 7 (2019), pp. 69226–69245. DOI: 10.1109/ACCESS.2019.2918469 (cit. on p. 9).

[Sal21] Salesforce. *Low Code: Das einfache Programmieren?* Feb. 3, 2021. URL: https://www.salesforce.com/de/blog/2021/02/low-code.html (visited on 09/30/2022) (cit. on p. 4).

[San15] S. Sandusky. “Gamification in Education”. In: The University of Arizona, 2015 (cit. on p. 10).

[SBB+22] S. Speth, S. Becker, U. Breitenbücher, P. Fuchs, N. Meißner, A. Riesch, D. Wetzel. “IT-REX — A Vision for a Gamified e-Learning Platform for the First Semesters of Computer Science Courses”. In: *Software Engineering im Unterricht der Hochschulen (SEUH 2022)*. Gesellschaft für Informatik, Bonn, 2022, pp. 43–48. DOI: 10.18420/SEUH2022_05 (cit. on pp. 1, 4, 11, 21).

[SGZ12] P. Sánchez, D. García-Saiz, M. Zorrilla. “Software product line engineering for e-learning applications: A case study”. In: *2012 International Symposium on Computers in Education (SIIE)*. 2012, pp. 1–6 (cit. on p. 12).

[SIDP20] A. Sahay, A. Indamutsa, D. Di Ruscio, A. Pierantonio. “Supporting the understanding and comparison of low-code development platforms”. In: *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. 2020, pp. 171–178. DOI: 10.1109/SEAA51224.2020.00036 (cit. on p. 9).

[SKRB22] S. Speth, N. Krieger, G. Reißner, S. Becker. “Teaching during the Covid-19 Pandemic — Online Programming Education”. In: *Software Engineering im Unterricht der Hochschulen (SEUH 2022)*. Gesellschaft für Informatik, Bonn, 2022, pp. 89–94. DOI: 10.18420/SEUH2022_09 (cit. on p. 11).

[SMYK14] G. Surendeleg, V. Murwa, H.-K. Yun, Y. S. Kim. “The Role of Gamification in Education – A Literature Review”. In: *Contemporary Engineering Sciences* 7.29 (2014), pp. 1609–1616 (cit. on p. 10).

[Uni12] University of Stuttgart. *Programmierung und Softwareentwicklung*. 2012. URL: https://www.student.uni-stuttgart.de/studienorganisation/modulhandbuecher/pdf/SoSe2012/mhb-81-909-1-2009.pdf (cit. on p. 11).

[Uni20a] University of Stuttgart. *Repository - Interactive Training Remote Education Experience*. 2020. URL: https://github.com/IT-REX-Platform (cit. on p. 11).

[Uni20b] University of Stuttgart. *Wiki - Interactive Training Remote Education Experience*. 2020. URL: https://github.com/IT-REX-Platform/Wiki/wiki (cit. on p. 11).

[Was19] R. Waszkowski. “Low-code platform for automating business processes in manufacturing”. In: *IFAC-PapersOnLine* 52.10 (2019). 13th IFAC Workshop on Intelligent Manufacturing Systems IMS 2019, pp. 376–381. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2019.10.060. URL: https://www.sciencedirect.com/science/article/pii/S2405896319309152 (cit. on p. 9).

[WFZS21] Y. Wang, Y. Feng, M. Zhang, P. Sun. “The Necessity of Low-code Engineering for Industrial Software Development: A Case Study and Reflections”. In: *2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. 2021, pp. 415–420. DOI: 10.1109/ISSREW53611.2021.00112 (cit. on p. 9).

All links were last followed on September 30, 2022.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

place, date, signature