TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Informatics

Anna Sooniste 183904IAIB
Ewert Ubaleht 193975IAIB
Ruslan Nesterov 155249IAIB

MOODLE AND DIGIKOGU INTEGRATION: DEVELOPING A MOODLE PLUGIN FOR SENDING GRADUATION THESES TO DIGIKOGU

Bachelor Thesis

Supervisor
Ago Luberg
PhD

Tallinn 2022

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Informaatika

Anna Sooniste 183904IAIB
Ewert Ubaleht 193975IAIB
Ruslan Nesterov 155249IAIB

MOODLE'I JA DIGIKOGU INTEGREERIMINE: MOODLE'I PISTIKPROGRAMMI ARENDAMINE LÕPUTÖÖDE SAATMISEKS DIGIKOKKU

Bakalaureusetöö

Juhendaja
Ago Luberg
PhD

Tallinn 2022

Author's declaration of originality

We hereby certify that we are the sole authors of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author 1:	Anna Sooniste	(digitally signed)
Author 2:	Ewert Ubaleht	(digitally signed)
Author 3:	Ruslan Nesterov	(digitally signed)

Date: May 30, 2022

Annotatsioon

Käesoleva bakalaureusetöö raames on loodud Moodle'i pistikprogramm lõputööde saatmiseks Moodle'ist Digikokku. Pistikprogrammi eesmärgiks on vähendada digikogustajate töömahtu, mis tuleneb lõputöö andmete ükshaaval Digikokku sisestamisest ning tagada platvorm, mis toetab lõputööde protsessi ning andmete ja failide jagamist ühes keskkonnas.

Pistikprogrammi puhul on tegemist Moodle'i Activity pistikprogrammiga, mis võimaldab Moodle'i kursuse eest vastutaval isikul ise hallata oma kursuse Graduation thesis pistikprogramme. Pistikprogramm on liidestatud Groups API-ga ning kasutab failide esitamiseks Assignment pistikprogrammi, et tagada plagiaadikontrolli võimalus.

Pistikprogrammi arendamisel on kasutatud PHP programmeerimiskeelt ning Moodle'is kasutusel olevaid API-isid, sealhulgas Groups API, Data Manipulation API, Enrollment API, File API, Form API, Access API and String API. Ühendus Moodle ja Digikogu vahel on loodud kasutades Digikogu Partner APId. Andmebaasiga suhtluse testimiseks kasutati projekti käigus MariaDB-d.

Töö tulemusena valmis Graduation thesis pistikprogramm, mida on võimalik Moodle'i kursuse omanikel lisada oma kurusele. Pistikprogramm võimaldab eri funktsionaalsuste täitmist vastavalt rollidele määratud õigustele.Pistikprogrammi administraatori seadetes saab muuta Digikogu Partner API seadeid ning ülikooli asutuste ja lõputöö tüüpide nimetusi. Pistikprogrammi mooduli seadetes on võimalik muuta lõputöödele külge lisatavaid üldiseid andmeid, sealhulgas instituuti, teaduskonda, õppekava ja lõputöö tüüpi, lisaks on antud vaates võimalik määrata grupeering, milles olevate autorite gruppide põhjal luuakse lõputööd ning Assignment pistikprogrammi moodulid, mida kasutatakse failide andmebaasist kätte saamiseks. Põhiprotsess hõlmab endas autorite poolt andmete sisestamist, muutmist ja kinnitamist, vajadusel juhendajate poolt juhendamise kinnitamist, ühe kuni kolme läbivaataja rolli poolt lõputööde läbivaatamist, kaitsmiskuupäeva sisestamist, vajadusel juurdepääsupiirangu määramist ning lõputööde Digikokku saatmist.

Lähtekood on leitav GitLabis:

`https://gitlab.cs.ttu.ee/ained/graduationthesis`.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 64 leheküljel, 8 peatükki, 20 joonist, 1 tabelit.

Abstract

As part of the graduation thesis in hand, a Moodle plugin for sending graduation theses from Moodle to Digikogu was built. The aim of the plugin is to decrease the workload of digital collectors. High workload is caused by the fact that the data of the theses must be inserted to Digikogu one by one. Additionally, the goal is to build a platform that supports the process and sharing data and files within one system.

The plugin created is a Moodle Activity plugin that supports the idea of allowing the manager of the course to manage the Graduation thesis plugins within the course. The plugin is integrated with Groups API and uses the Assignment plugin for files submission to allow plagiarism checks.

For the development of the plugin PHP programming language was used and the APIs of Moodle, including Groups API, Data Manipulation API, Enrollment API, File API, Form API, Access API and String API. Connection between Moodle and Digikogu was created by using Digikogu Partner API. For testing the communication with the database, MariaDB was used during the project.

As a result of the project Graduation thesis plugin was created. The plugin can be added to Moodle courses by the course managers. The plugin supports functionalities according to the permissions set for roles. In the administrator settings of the plugin the Digikogu Partner API details can be changed but also the names of the institution, schools and theses types. In the settings of the plugin module it is possible to change the data that is attached to the theses within the plugin module. The data includes the institution, school, study programme and the type of the thesis. Additionally, it is possible to define the grouping, which includes the groups based on which the theses are created. Assignment plugin modules that are used for the Graduation thesis plugin are also defined in the module settings view. The main process includes authors inserting, updating and confirming the data and files; if needed, supervisors confirming supervision; one to three roles reviewing the theses; inserting defence dates; if needed inserting access restrictions and sending theses to Digikogu.

Source code is accessible at GitLab: `https://gitlab.cs.ttu.ee/ained/graduationthesis`.

The thesis is in English and contains 64 pages of text, 8 chapters, 20 figures and 1 table.

List of abbreviations and terms

Digikogu	Digital library of the Tallinn University of Technology
Processor	Graduation thesis topics management system
Moodle	Open-source learning platform
SIS	Study information system
Digital collector	Employee of the university who inserts theses to Digikogu

Table of Contents

List of Figures ix

List of Tables x

1 Introduction 1
- 1.1 Problem Statement . . . 1
- 1.2 Task and goals setting . . . 2

2 Project Description 4
- 2.1 Use cases . . . 5
  - 2.1.1 Repeating processes in use cases . . . 9
  - 2.1.2 USE CASE 1: Students submit and digital collectors send theses to Digikogu . . . 10
  - 2.1.3 USE CASE 2: Students submit, supervisors review and digital collectors send theses to Digikogu . . . 11
  - 2.1.4 USE CASE 3: Students submit, supervisors review, evaluation committee reviews and adds defence dates, digital collectors send theses to Digikogu . . . 12
- 2.2 Existing work . . . 13

3 Project Design 15
- 3.1 Backend . . . 15
  - 3.1.1 API-s . . . 15
  - 3.1.2 Digikogu partner API . . . 18
  - 3.1.3 Validation . . . 18
  - 3.1.4 Backend design . . . 18
  - 3.1.5 Database design . . . 20
- 3.2 Front end . . . 22
  - 3.2.1 User interface design . . . 23
- 3.3 Code design . . . 23

4 Project Contents 24
- 4.1 Backend . . . 24
  - 4.1.1 Assignment module . . . 24
  - 4.1.2 Moodle APIs . . . 24
  - 4.1.3 Digikogu partner API . . . 39

4.1.4 Validation . . . 41
4.2 Frontend . . . 42

5 Analysis **52**
5.1 Process overview . . . 52
5.1.1 Data types . . . 52
5.1.2 Data insertion stages . . . 52
5.1.3 Data presentation . . . 53
5.1.4 Main view data . . . 53
5.1.5 Availability of functionalities . . . 53
5.1.6 Confirmations process . . . 54
5.1.7 Supervision confirmation . . . 55
5.1.8 Views versus capabilities . . . 55
5.1.9 Assignment plugin . . . 56
5.1.10 Groups API . . . 56
5.2 Valuable meetings . . . 56
5.2.1 26th January: Meeting with all of the clients . . . 56
5.2.2 2nd February: Meeting with stakeholders . . . 57
5.2.3 22nd April: Demo to clients and couple of digital collectors . . . 57

6 Results and Validation **58**

7 Comments **60**

8 Conclusion and Future Development **62**

Bibliography **65**

Appendices **68**

Appendix 1 - Database Diagram **68**

Appendix 2 - Rules for Publication and Preservation of Graduation Thesis **71**

List of Figures

1 Backend folder 19
2 Project centered database diagram highlighted 22

3 Digikogu Partner API Flow Chart 40
4 Main view with a freshly created thesis 43
5 Main view with the filtering option 44
6 Supervision confirmation 44
7 Main view with extra button layer 45
8 Thesis collapsed overview 45
9 Thesis view action buttons 45
10 Author(s) section overview 46
11 Supervisor(s) section overview 46
12 Submitted thesis view data section 48
13 Submitted thesis view administrative data section 49
14 Thesis form overview 49
15 Author(s) input 50
16 Supervisor(s) input 50
17 Thesis data input 51

18 Database diagram highlighted 69
19 Project centered database diagram 70
20 Project centered database diagram highlighted 71

List of Tables

1 *Allocation of capabilities* 32

1. Introduction

At most of the universities including the Tallinn University of Technology, as the final stage of the studies, students write graduation theses. While back in the days, graduation theses were submitted on paper, with the aim to 'redesign the work processes of the university by using new, evolving and breakthrough technologies to make the university the most well-digitized campus in the world' [1], the Tallinn University of Technology has moved to storing theses digitally. For this, Digikogu - a digital library, has been created. While the digital environment exists, integration with study platforms and study information system is non-existent. As a result the workload of the employees of the university's schools that are managing the theses has increased. The project in hand focuses on integrating Moodle - an open-source learning platform and Digikogu.

1.1 Problem Statement

Today, digital versions of graduation theses are sent to Digikogu by inserting the details of the theses one by one by the employees of different schools of the university, hereinafter referred to as digital collectors. In the first and second level of higher education, there are approximately 2000 graduation theses written a year at Tallinn University of Technology. According to the 2021 academic year statistics, the study programs had a maximum of 156 graduates per program.[2]

Considering the amount of graduation theses and the data that has to be inserted and published at Digikogu, the process is time-consuming. According to a study conducted on human errors in data entry, only 5.5% of the research participants that were tested in single entry of data had perfect accuracy. [3] Thus, the likelihood of a digital collector making a mistake in data entry is rather high. Furthermore, the theses data and files are shared between a range of parties including students, supervisors, evaluation committee members and digital collectors by using a range of communication channels and are stored in multiple locations that the university does not have a complete overview of.

As a result of a study conducted in 2021 among digital collectors to gain better understanding of the processes of different study programs, it was found that Moodle is one of the most common locations for storing the theses. To be more precise, data was collected

about 61% of the study programs and of those 36% store graduation theses in Moodle.[4] As a result of this information a problem was identified and a course for a solution also set.

1.2 Task and goals setting

It is clear that the process of sending theses to Digikogu can and should be automated by using the data and files that students submit for final defence of graduation theses. However, there are a few constraints to that happening fluently. Today, the submission of the metadata of the graduation theses takes place in the study information system of Tallinn University of Technology. The same metadata is needed for Digikogu and is therefore inserted by hand by digital collectors. In addition, the files are not entered to the study information system but shared between parties depending on how the process is built for a study program.

As it was identified, Moodle is one of the most commonly used platforms at Tallinn University of Technology for storing and managing the processes and files of graduation theses. Furthermore, Moodle supports the plagiarism detection system Ouriginal/Urkund [5], which is used by the university. All of the theses must go through plagiarism detection system before they can be sent to Digikogu.

The perfect solution would be to get data both from SIS - the study information system of the university and from Moodle to send graduation theses to Digikogu, however at the moment of conducting the project, the two systems have not been integrated and therefore, the idea cannot be implemented.

Moodle is chosen as the platform to develop the solution on. First of all, it is an open-source learning platform that allows the authors of the project to easily start off the development process. Secondly, it is already used for part of the process by students, teachers and other employees of the university. This solution is created with the possibility in mind to integrate SIS and Moodle in the future to avoid repetitive data insertion and reviewing. Currently however, the project focuses only on building the whole process in Moodle for data and files submission, the reviewing process and the process of sending submitted and reviewed theses to Digikogu.

The project aims to use as much data that already exists in the Moodle database as possible to decrease the probability of mistakes happening. Furthermore, it aims to make it possible for supervisors and evaluation committee to be involved in the process and to review the data and files of the theses to decrease the dependency of correct data on digital collectors and with that also decrease the likelihood of mistakes taking place. Additionally,

probability of mistakes is decreased by bringing attention to possible mistakes in data and missing data by using data validation before any submissions take place.

Another goal is to make students take more responsibility of their data being correct. This can be achieved by adding check-boxes and alerts for students to confirm that the data inserted is in line with the requirements.

To conclude, the goal is to build a plugin in Moodle that supports students submitting the final data and files of their graduation theses and secondly, students taking more responsibility in submitting correct data and files. The solution allows

to include other parties than digital collectors and students to be engaged in the reviewing process,
reviewers to change data and files if there are mistakes done
to provide feedback/comments for theses,
to return student confirmations of data and files submission to allow students to make additional changes
to give confirmations of the data being reviewed
digital collectors to send reviewed theses to Digikogu
to decrease the workload of digital collectors
to decrease the probability of mistakes in theses sent to Digikogu

2. Project Description

The graduation thesis of sending graduation theses from Moodle to Digikogu can be split into three important stages.

1. Analysing the business process and communicating with the client
2. Development
3. Final analysis and writing the graduation thesis final paper

To start off with the development process as quickly as possible during the first months roles were split as follows:

> Anna Sooniste: Analysing and concluding the process that needed to be developed. Communicating with the client to get as many answers that would narrow down the scope of the project. Support the learning process of co-authors getting to know Moodle and the process that needed to be developed as a result. Setting up the initial plugin.
> Ewert Ubaleht and Ruslan Nesterov: familiarizing themselves with Moodle Developer Documentation and possibilities. Kick-starting the development process.

Once all of the authors were on the same page with both Moodle development possibilities and with the client-side needs, the full-scale development stage, engaging all of the three authors, started.

Client-side of the project can be split into two. First, the Office of Academic Affairs of Tallinn University of Technology, represented by Liisi Järve and Anna Varkki from the Educational Technology Centre and by Katrin Valvik from the Software Development Division. Second, the School of Information Technologies represented by Ago Luberg, the Programme Director of Informatics, who is also the supervisor of the thesis in hand.

It is important to note that during the project we have considered the feedback of

> the Educational Technology Centre who will be responsible of the plugin in the future.

the university's employees whose job is to insert theses to Digikogu and who will be the key users of the plugin in the future.
Ago Luberg who has extensive experience in the role of a supervisor, an evaluation committee member, a teacher and has had to work with many Moodle courses.

Due to the complexity of the project there was clear need for a platform to support the management of the project identified. As the project is focused on supporting a process of the Tallinn University of Technology and all the stakeholders are from within the company, GitLab - a tool that is commonly used by students, the School of Information Technologies and the university's IT department, was selected.

For communication it was decided to use Teams, a platform that all of the relevant parties have access to. Two channels were created to make sure everybody engaged are in the same information space - one to support the communication within the team and with the supervisor, second to support the communication with the client side (Office of Academic Affairs).

Requests and information from the client-side (Office of Academic Affairs) and tasks to develop were discussed and agreed on at weekly meetings. Thesis' supervisor, who was also in a client role, joined the meetings every second week.

2.1 Use cases

Graduation thesis plugin is created keeping in mind the complexity of the processes while imposing the regulations of the university. The plugin is created for Tallinn University of Technology, and the aim was to make the plugin as flexible as possible to support the differences in the processes among the schools and study programs of the university.

Plugin settings

Plugin settings (possible to edit only by the company's administrator of Moodle) make it possible to change the Digikogu URL, Digikogu API-Key, name of the institution, schools of the institutions, and graduation thesis types that the plugin is used for.

This means that if the name of the university, schools, thesis types, Digikogu URL or API-key should change, the plugin can still be used with the new data by inserting different values to the named fields in the plugin settings.

Key functionalities: permissions/capabilities

Within Tallinn University of Technology the processes in different schools vary and finding a common ground for an IT solution seems to be more time-consuming than creating a solution with a range of possibilities. Therefore, when generating the solution, the focus was more on the School of Information Technologies needs, nevertheless on every step of the way flexibility was in mind to make sure that if another school wanted to use the plugin they would have as many possibilities to make the plugin work according to their needs. As a result we have created a plugin that is built on capabilities that can be given to different roles by the manager of the course.

The key functionalities in the process of sending theses to Digikogu that are built as capabilities/permissions [6] can be split into five groups:

1. Administrative capabilities
   (a) Permission to add plugin instances to a course (Add instance capability)
2. Viewing permissions
   (a) Permission to view only own theses and access a plugin instance (View capability)
   (b) Permission to view only theses under user's supervision (View own students capability)
   (c) Permission to view all theses in plugin (View all capability)
3. Confirmation permissions (further explanation provided below)
   (a) Author confirmation permission (Author confirmation capability)
   (b) First confirmation permission (First confirmation capability)
   (c) Second confirmation permission (Second confirmation capability)
   (d) Final confirmation permission (Final confirmation capability)
4. Permissions dependent on confirmation permissions
   (a) Permission to change the metadata and files of theses (Update thesis data capability)
   (b) Permission to give editing permission back to the author (Re-enable author confirmation capability)
   (c) Permission to create a new thesis (Submit capability)
5. Other independent permissions
   (a) Permission to give feedback (Give feedback capability)
   (b) Permission to insert defence date (Insert defence date capability)
   (c) Permission to insert access restriction (Insert access restriction capability)
   (d) Permission to confirm supervision (Supervision confirmation capability)
   (e) Permission to send theses to Digikogu (Send thesis to Digikogu capability)

Viewing permissions define which theses can be seen by the role but also which theses can be edited by using rest of the capabilities.

Confirmation permissions are for defining which roles and in what order they can confirm theses, however they also define the possibility to use permissions dependent on confirmation permissions. The order of the confirmation permissions is the following:

1. Author confirmation permission
2. First confirmation permission
3. Second confirmation permission
4. Final confirmation permission

Author and final confirmation permissions must be defined for roles for the process to work. First and second confirmation permissions are additional permissions for cases when more parties than authors and digital collectors take part in the reviewing process. If one of the confirmation permissions is left out, the process will jump over that stage.

Other independent permissions simply apply when capability is assigned to a role.

Permissions dependent on confirmation permissions can only be used when a role is provided with the dependent permission but also with with a confirmation permission as follows:

1. Author confirmation permission
    (a) Permission to change the metadata and files of theses
2. First confirmation permission
    (a) Permission to change the metadata and files of theses
    (b) Permission to give editing permission back to the author
3. Second confirmation permission
    (a) Permission to change the metadata and files of theses
    (b) Permission to give editing permission back to the author
4. Final confirmation permission
    (a) Permission to change the metadata and files of theses
    (b) Permission to give editing permission back to the author

Assignment plugin and submitting files

Before a plugin instance is added it is required to create up to three assignment modules for submitting 3 types of files: main graduation thesis file, summary of graduation thesis

and appendices. Either one assignment can be made for submitting all the files or up to three assignments to separate the files in the Graduation thesis plugin instance.

Grouping and groups

Graduation thesis plugin is integrated with Moodle built-in functionalities of groups and groupings [7]. Every single thesis visible in Graduation thesis plugin is connected to a group in a grouping that is connected to the Graduation thesis plugin instance.

> If a new group is added to the specific grouping of the thesis plugin instance, a new thesis is also created.
> If a new thesis is added in the Graduation thesis plugin instance, a new group is added to the grouping.
> If a group is deleted, the thesis is deleted.
> If an author is removed from a group, the person is also removed from the thesis data.

Individual theses

For creating an individual thesis, author must simply add submission in the plugin instance and a new group is created in the Graduation thesis plugin instance, as a result also a new thesis is created.

Group theses

For creating a group thesis, authors must be in the same group of the Graduation thesis plugin instance grouping. There are two recommended ways for adding authors to groups, however other plugins that are connected to the Groups API can also be tested.

1. Course manager can add groupselect plugin [8], where students can arrange their groups by themselves, to the course or integrate other plugins that are connected to groups.
2. Course manager can arrange students into groups in the Groups view of the course.

Creating groups can be done both in advance or after the plugin is already created by using the grouping defined in the Graduation thesis plugin instance settings.

Use cases heavily depend on which capabilities are given to which roles. The following recommended use cases are explained by using the roles that exist in Moodle on initial install as examples, however can also be selected differently. The focus of the use cases

should be on how capabilities are divided among roles and grouped together not on which role has which capability.

It is also important to take notice of the fact that the use cases explained below are not written to restrict the possibilities of the plugin in any way but simply a few examples, how the capabilities of the plugin can be used.

The common process before every use case:

1. Administrator of Moodle updates the plugin settings.
2. Course manager creates assignment modules for files submissions
3. (Not required but recommended) If group theses are expected, course manager creates a grouping and either adds groups and groups' members or adds the Group self-selection plugin to the course and allows students to create groups by themselves.
4. Course manager adds Graduation thesis plugin to the course and at the plugin instance settings inserts the required information, including institution, school of the institution, program name and code, graduation thesis type, assignment modules used for files submissions and the grouping (if not selected is created automatically).

2.1.1 Repeating processes in use cases

Author confirmation and supervision confirmation (example roles: Student, Teacher and Manager)

1. Managers can see all the theses created with the data provided in plugin instance settings and groups.
2. Authors can see their own theses and make changes to the metadata and files. They select a supervisor 1 and a supervisor 2 from course members that are not in student roles (supervisor details are then taken from Moodle database) or they insert supervisors details by hand.
3. One of the authors saves the data and files of the thesis.
4. If a supervisor 1 or a supervisor 2 is selected from the course members, the supervisor must confirm supervision of the thesis. If supervisor details are inserted by hand, this step is skipped.

Final confirmation, adding defence date and access restriction, sending theses to Digikogu (example role: Manager)

1. Managers receive the permission to confirm the thesis. With confirmation permission Managers also have the permission to make changes to metadata and files or the permission to give confirmation permission back to authors.
2. One of the Managers adds defence date to the thesis.
3. If needed one of the Managers adds access restriction to the thesis.
4. After Managers have made changes or the confirmation permission has returned from Student to Manager, one of the Managers confirms the thesis.
5. One of the Managers makes sure that all the data is correct, that defence date is inserted and sends the selected thesis to Digikogu.

First confirmation (example role: Teacher)

1. Depending on the viewing permission:
   a) **Permission to view only theses under user's supervision:** Teacher who has confirmed supervision can see the confirmed thesis, make changes to it, give the permission to make changes and confirm the thesis back to Students, confirm the thesis. In case there is no Teacher that has confirmed supervision - supervisor data was inserted by hand, the confirmation permission will jump over to final confirmation permission (Managers: digital collectors).
   b) **Permission to view all theses in plugin:** Teachers receive the permission to confirm the thesis. With confirmation permission Teachers also have the permission to make changes to the metadata and files or the permission to give confirmation permission back to authors.
2. After Teachers have made changes or the confirmation permission has returned from Students to Teachers, one of the Teachers confirms the thesis.
3. After one of the Teachers has confirmed the thesis, Teachers will not be able to make any changes.

2.1.2 USE CASE 1: Students submit and digital collectors send theses to Digikogu

Roles and capabilities:

Student: permission to view only own theses; permission to change the metadata and files of theses; author confirmation permission;
Manager: permission to view all theses in plugin; permission to change the metadata and files of theses; final confirmation permission; permission to give feedback;

permission to give editing permission back to the author; permission to send theses to Digikogu; permission to insert defence date; permission to insert access restriction

Overview

This use case is the simplest and good for processes which only include students submitting their data and digital collectors reviewing and sending the theses submitted to Digikogu.

1. See author confirmation and supervision confirmation (example roles: Student, Teacher and Manager) process at 2.1.1.
2. After one of the authors has saved the thesis data and supervisors have confirmed supervision, one of the authors will confirm the thesis. Now, authors will not be able to make any changes unless they are given the permission for it by Managers.
3. See final confirmation, adding defence date and access restriction, sending theses to Digikogu (example role: Manager) process at 2.1.1.

2.1.3 USE CASE 2: Students submit, supervisors review and digital collectors send theses to Digikogu

Roles and capabilities:

Student: permission to view only own theses; permission to change the metadata and files of theses; author confirmation permission;
Teacher: permission to view only theses under user's supervision*; permission to change the metadata and files of theses; first confirmation permission; permission to give editing permission back to the author; permission to give feedback
Manager: permission to view all theses in plugin; permission to change the metadata and files of theses; final confirmation permission; permission to give feedback; permission to give editing permission back to the author; permission to send theses to Digikogu; permission to insert defence date; permission to insert access restriction

* By giving the teacher the permission to view all theses in plugin, teachers can review all theses and be used as overall reviewers (not only as reviewers of the theses under their supervision) before reviewing process goes over to digital collectors.

Overview

This use case is good for cases when supervisors are expected to review the metadata and files before digital collectors can review the data and send theses to Digikogu.

1. See author confirmation and supervision confirmation (example roles: Student, Teacher and Manager) process at 2.1.1.
2. After one of the authors has confirmed the thesis and supervisors have confirmed supervision, authors will not be able to make any changes unless they are given the permission for it by Teachers or Managers.
3. See First confirmation (example role: Teacher) process at 2.1.1.
4. See final confirmation, adding defence date and access restriction, sending theses to Digikogu (example role: Manager) process at 2.1.1.

2.1.4 USE CASE 3: Students submit, supervisors review, evaluation committee reviews and adds defence dates, digital collectors send theses to Digikogu

Roles and capabilities:

Student: permission to view only own theses; permission to change the metadata and files of theses; author confirmation permission;
Teacher: permission to view only theses under user's supervision*; permission to change the metadata and files of theses; first confirmation permission; permission to give editing permission back to the author; permission to give feedback
Non-editing teacher: permission to view all theses in plugin; permission to change the metadata and files of theses; first confirmation permission; permission to give editing permission back to the author; permission to give feedback; permission to insert defence date
Manager: permission to view all theses in plugin; permission to change the metadata and files of theses; first confirmation permission; second confirmation permission; final confirmation permission; permission to give feedback; permission to give editing permission back to the author; permission to send theses to Digikogu; permission to insert defence date; permission to insert access restriction

* By giving the teacher the permission to view all theses in plugin, teachers can review all theses and be used as overall reviewers (not only as reviewers of the theses under their supervision) before reviewing process goes over to evaluation committee.

Overview

This use case is good for cases when supervisors are expected to review the metadata and files, then evaluation committee members are expected to review the theses and insert defence dates and finally, digital collectors can review the data and send theses to Digikogu.

1. See author confirmation and supervision confirmation (example roles: Student, Teacher and Manager) process at 2.1.1.
2. After one of the authors has confirmed the thesis and supervisors have confirmed supervision, authors will not be able to make any changes unless they are given the permission for it by Teachers, Non-editing teachers or Managers.
3. See First confirmation (example role: Teacher) process at 2.1.1.
4. Non-editing teachers receive the permission to confirm the thesis. With confirmation permission Non-editing teachers also have the permission to make changes to the metadata and files or the permission to give confirmation permission back to Students.
5. Non-editing teachers add defence date to the thesis.
6. After Non-editing teachers have made changes or the confirmation permission has been returned from Students to Non-editing teachers, one of the Non-editing teachers confirms the thesis.
7. After a Non-editing teacher has confirmed the thesis, Non-editing teachers will not be able to make any changes.
8. See final confirmation, adding defence date and access restriction, sending theses to Digikogu (example role: Manager) process at 2.1.1.

2.2 Existing work

One of the authors of the project worked with another team on the same problem during the internship in the summer of 2021. The internship resulted with a prototype that was further developed by the Software Development Division of the university, nevertheless it never went live due to many reasons. The prototype however provided the authors of the thesis in hand with a great insight. Some of the key takeaways from that project are:

1. that the Digital Collector should be able to edit data before sending it to Digikogu.
2. to present validation information in the form not in a pop-up alert.
3. to present dates in the form of dd.mm.yyyy
4. validate data before any submissions
5. that using Questionnaire plugin caused problems and it might be beneficial to avoid using it and develop a form strictly meant for the theses' data insertion.
6. to keep track of the requests to and responses from the Digikogu Partner API.
7. that access restrictions should be added to the theses by employees (Digital Collectors) not by authors.
8. that instead of adding the final date of the access restriction provide options for setting the access restriction for 3 years and 5 years that is then automatically calculated by adding the years on top of the defence date.

While the prototype was used to generate ideas and avoid making the same mistakes again, the project in hand has been built as a completely separate project.

3. Project Design

Our application is written onto Moodle Learning Platform and course management system (CMS). Moodle is one of the most widely used course management systems, with a flexible foundation. Moodle is a free and open source course management system. With plug-ins, activity modules, blocks, and other features, anyone may easily contribute to the development of Moodle. Moodle allows instructors, administrators and students to make individualized learning environments using a single, secure, and integrated system.

Activity modules are the most popular method of contributing. By default, Moodle offers a list of activities (forums, quizzes, assignments, etc.). Activity module method was chosen for the Graduation thesis plugin. The reasoning behind this is that activity modules are one of the most important types of Moodle plugins since they provide activities in courses. For instance, there's a Forum, a Quiz, and an Assignment. The Graduation thesis plugin needs for example the Assignment activity for adding theses files. It was important that course managers could create this plugin when creating a course or add it even if the course is already in use. [9], [10]

3.1 Backend

The server-side of a website is referred to as the backend. It organizes and stores data to database, as well as ensures that everything on the client side of the website functions properly. It is the section of the website you can not see or interact with. It is the part of the software that does not interact with users directly. Users have indirect access to the pieces and attributes created by backend designers via a frontend application. [11]

In our case the backend application is built using PHP programming language. PHP is a server-side programming language that was created with web development in mind. PHP is known as a server-side scripting language since it executes code on the server. PHP is used in this project because that is the language in which Moodle is developed in.

3.1.1 API-s

This chapter describes API-s used in the Graduation thesis plugin.

Groups API

In Moodle, groups are collections of users in a course. They can be made manually during a bulk user upload or designated by the teacher in the course settings (eg from a text file). Groups can be joined together into named Groupings if desired.

A teacher can choose whether or not a course utilizes groups and whether or not such groups are separate (users can only see people in their own group) or visible in the course settings (can see other groups just like teachers). This setting can be applied to all course activities or not and thus, giving a lot of flexibility. [12]

The Groups API was decided to be used for theses that have more than one author to make the plugin more user friendly. This API is already used for example in the Moodle course for graduation theses of the Informatics study program. Using this API does not prevent authors from entering thesis alone.

Data manipulation API

This API includes functions for retrieving or altering database items. These functions provide a high level of abstraction and ensure that the database manipulation will operate across several RDBMSs. [13]

In the Graduation thesis plugin this API is used for saving and retrieving data from database.

Enrollment API

Enrollment API can be used for example for getting all the enrolled users or for determining whether they are enrolled to a specific course or not. [14]

In the plugin created, the API is used for displaying users in drop down inputs. This way users can choose supervisors that already have existing Moodle accounts.

File API

The File API is used to manage all of Moodle's files. [15]

File API is used in the project for displaying thesis, summary and appendices files for the users, so then they can download those files.

Form API

The Form API is used to generate web forms in Moodle. All HTML elements (checkboxes, radio buttons, text boxes, and so on) are supported by the Form API, which improves accessibility and security. [16]

Form API was decided to be used for all the forms in the plugin because it is built in Moodle, it is easy to use and it had all the necessary elements needed for the forms that were built.

Access API

The Access API provides functions for determining the current user's permissions. It also allows modules to add new capabilities to Moodle.

Moodle's access control system is based on roles. In Moodle, the majority of elements (system, users, course categories, courses, modules, and blocks) are represented by contexts, which are organized in a tree-like hierarchy called the context tree. A role is a collection of capability definitions, each of which typically represents a user's capability to do a specific task. Roles are specified at the highest level of the system context hierarchy. At lower context levels, role definitions can be overridden. The definitions of roles assigned to users are used to determine user access control. [17]

This API is used to create the essential capabilities of the Graduation thesis plugin. This API supports the flexibility to build personalised processes among various study programs. Each course manager can add and remove the capabilities to/from roles that are necessary for the particular institution, faculty, curriculum or course.

String API

String API is used to present strings in the user interface according to the language Moodle is used in. It takes care of internationalization problems and will offer the best text to each user based on a variety of settings and environment variables. Moodle features a mechanism that allows language strings to be found in a number of places (in order). This allows language strings to be packed with plugins, eliminating the need to copy language files to the language directory when installing a plugin. [18]

There two different languages in the Graduation thesis plugin: English and Estonian.

3.1.2 Digikogu partner API

Digikogu Partner API is an intermediary of the Digikogu that allows to use the `POST`, `PUT` and `GET` requests for uploading graduation thesis to Digikogu. The process of uploading thesis to Digikogu by using the Digikogu Partner API has 5 stages.

1. Creating a folder for a theses by attaching the metadata of the theses to the folder. `POST` request is used here.
2. Creating folders inside the theses folder for files. `POST` request is used here.
3. Adding files to the folders. `PUT` request is used here.
4. Publishing the theses folder. `PUT` request is used here.
5. Making sure the theses folder is published. `GET` request is used here.

Every time a request is made a response is also received which provides information about the details of the theses folders but also about whether the request was successful or not. [19]

To be able to make requests to the API, a specific URL and api-key must be used. The Tallinn University of Technology has two URL-s for the API - one for the public Digikogu and second for the Digikogu testing environment. This project was developed by using the details of the testing environment.

3.1.3 Validation

User input is validated when new thesis data and files are inserted. The data validated is for example keywords and e-mail but it is also checked, whether all the required fields are entered, including authors' matriculation numbers and a primary supervisor. When a user confirms their thesis, it is made sure that all the data is present.

3.1.4 Backend design

Activity modules reside in the `/mod` directory. Moodle requires a number of files to function properly. These files are used to install and integrate your module into the Moodle system. Each file serves a specific purpose; some files are unnecessary and are only created when the functionality they provide is required. The key files that are used are `db/access.php`, `db/install.php`, `db/install.xml`, `db/upgrade.php`, `lang/est/graduationthesis.php`, `lang/en/graduationthesis.php`, `version.php`.

`db` folder contains files like `access.php`, where the capabilities of the plugin are defined. `install.xml` file defines the database tables. `upgrade.php` file is responsible for updating the module to the most recent version. `lang` folder contains translation files. The project has Estonian and English translation files. `classes` folder holds Moodleform classes that are used for displaying different forms like thesis form, where users can change their thesis data and files. `pix` folder contains icon that Moodle displays next to the name of the plugin. `version.php` file is used for keeping track of the version of the module as well as other information such as the Moodle version it requires. Folder structure is displayed in figure 1.

Some of the files that are also important in the Graduation thesis plugin are named below. `mod_form.php` file is used when adding/editing a module to a course. It contains the components that will appear on the form that will be used to create/install a module instance. Moodle uses `index.php` page to list all of the instances of a module that are in a specific course, with the course id given to this script. `view.php` file is the plugin's main page where theses table is displayed.

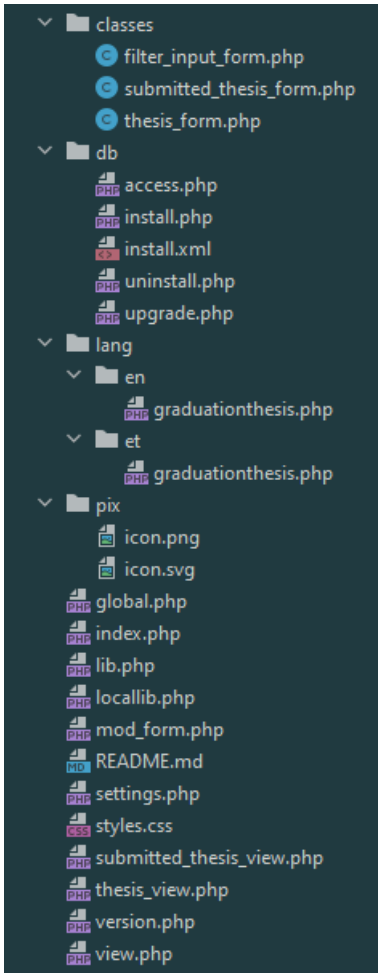

Figure 1. Backend folder

3.1.5 Database design

Moodle supports multiple databases such as MySQL, SQLite, Oracle, MSSQL and PostgreSQL.

Oracle was ruled out because it is not really recommended as it is generally slow and basically no core Moodle developers utilize it for development. MSSQL was also not in the consideration because there is no native support for MSSQL driver for PHP 7, which can add some unnecessary challenges. SQLite is simply not a client-server database engine, so that was ruled out as well.

For selecting the database to test the Moodle platform on, the decision was made between MySQL and PostgreSQL. Main reason for choosing MySQL/MariaDB was that MySQL community is very large, there are a lot of people that can help with it and MariaDB because it is considered to be more open and you can use it with any Moodle version that is stable and supported. [20]

All of the plugin related tables were created following the format of existing plugins and are described in `install.xml` file in the `db` folder. All the tables were created in the Moodle database. Table names follow the schema of having a `mdl` prefix added to the table name. Moodle automatically adds this prefix to the tables, when tables are created or accessed. The `install.xml` file has the description for total of 9 tables.

ERD Schema

As mentioned before, 9 tables were created in order to store graduation thesis related data. The structure of the database schema of the tables that were created is described in figure 2.

1. **graduationthesis** - Table that stores data about each plugin instance that has been added to course(s) with information such as assignment id-s, department, institution, curriculum codes etc.
2. **graduationthesis_data** - Table that stores information about a single graduation thesis submission with information such as title, language, keywords, defence date and access restriction, also thesis and summary file id-s in the files table that already existed in the Moodle database.
3. **graduationthesis_lang_pair** - Table that was created to store information about institutions, thesis types and departments including both their Estonian and English translations.

4. **graduationthesis_appendix** - Table that stores information regarding extra files submitted per thesis. Each extra file is it's own record.
5. **graduationthesis_json** - Table that contains data about all the requests made to the Digikogu Partner API.
6. **graduationthesis_api** - Table that holds the information about the Digikogu id-s of the theses that have been sent to Digikogu.
7. **graduationthesis_user** - Table that contains data about users who are involved with a specific thesis, such as first name, last name, e-mail, role, and matriculation number if they are an author.
8. **graduationthesis_feedback** - Table that stores information about the given feedback, who and when gave it.
9. **graduationthesis_confirm** - Table has information about all the confirmations with who and when gave it.

Figure 2. Project centered database diagram highlighted

3.2 Front end

The front end of a website is the part that the user interacts with directly. It's also referred to as the application's client side. Everything that users see directly: text colors and styles, photos, graphs and tables, buttons, colors, and the navigation menu. The languages used

for front end development are HTML, CSS, and JavaScript. Front-end developers provide the structure, appearance, behavior, and content of everything that appears on browser displays when websites, online applications, or mobile apps are opened. [11]

3.2.1 User interface design

User Interface Design is concerned with anticipating what users may need to do and ensuring that the interface contains features that are simple to access, understand, and use in order to assist those actions. Interaction design, graphic design, and information architecture are all combined in user interface.

Since a Moodle plugin was being built, there were not many options to design the web pages. Minimalistic design was used so all the parties interested in using this plugin could add their own styles and designs. For the forms the Form API that is built into Moodle was used. The theses' table and buttons used with the table are specially designed with HTML and small amount of CSS.

3.3 Code design

PHP code follows a few guidelines as for example all variable names, file names, translations and function names use snake_case, meaning they are written lowercase and with a underscore between words. Class names use PascalCase, where each new word starts with uppercase letter and all the words are written together. [21] Functions are in a `locallib.php` file to make code easier to use, inline CSS is avoided. HTML and JavaScript are in the PHP code files.

4. Project Contents

This chapter describes the implementation details of the project.

4.1 Backend

The Backend of the project handles the process of storing data to and getting data from required tables within the Moodle database.

4.1.1 Assignment module

In order to provide users a way to submit thesis related files, Assignment module was used in the thesis submission system. Assignment module can be configured to have the plagiarism plugin enabled. The Graduation Thesis plugin enables the Administrator or Course creator to configure which Assignment module instances are used for what type of files during thesis submission. This makes the developed plugin flexible by allowing the course creator to either have an Assignment module instance for each type of file or they can add a single Assignment module to the course and enable multiple file submissions on it. If the latter is chosen, the developed plugin instance must be configured to read the files from the same assignment module for each file type. When authors submit their thesis, all the files submitted by thesis' authors will be presented to choose from in the form.

Students have 3 types of files to submit - thesis file, summary file and appendices. Reference to the submitted thesis and summary files is saved to the plugin's `graduationthesis_data` table. Reference to the submitted extra files are saved to the `graduationthesis_appendices` table in the Moodle database making it easier to access and check for file presence. 8

4.1.2 Moodle APIs

Groups API

The developed plugin heavily relies on Moodle groups and groupings, thus, this API is necessary to ease the process of having the required data available to the plugin at all times.

Firstly, the Groups API is used to get the list of existing groupings on the current Moodle Course. The list of existing groupings is then used to provide the course creator a choice between the groupings during plugin addition to the course and configuration.

In order to connect a grouping to a graduation thesis instance in Moodle, an existing grouping can be selected on plugin instance addition, or a new grouping will be created automatically by the plugin if no grouping was selected. This grouping will by default have the plugin instance's name as its name to make it easier to find among existing groupings.

The configured grouping is not deleted on plugin instance deletion in order to prevent any data loss.

Theses data is created or updated each time `view.php`, `thesis_view.php` or `thesis_submitted_view.php` are opened. The theses are created or updated based on the updates made to the groups in the grouping connected to the instance of the plugin. New authors are added if a group received new members and removed if any member was removed from a group.

When a submitter, who is not part of a group of the grouping of the plugin's instance wants to submit a thesis, the group is automatically created with the current user and a relation between thesis and the group is made.

Data manipulation API

Moodle has a built-in API that provides easy access to all the necessary SQL requests. Functions used in the project are:

get_records() - used to get list of elements from the specified table by the provided search terms
get_records_list() - used to get list of elements from the specified table where the specified field matches any of the provided criteria
get_record() - used to get a singular element from the specified table
insert_record() - used to save a singular record to the specified table
insert_records() - used to insert multiple records at the same time to the specified table
update_record() - used to update a record in a specified table
delete_records() - used to delete list of elements from a specified table
record_exists() - used to check if a record with given search terms exists in a specified table

In order to write a custom SQL query, a function **get_record_sql()** was used. In one case it was used in order to get the latest confirmation made for a thesis. This allowed to get the data, sort it by id in descending order and get the first record. This allows to get the data via Backend instead of later filtering it in the Frontend. The other case where this function was used is when theses are filtered and sorted by certain criteria. This allowed to implement both features in a single query.

Enrollment API

In order to get the list of possible choices for thesis supervisors Enrollment API's get_enrolledusers is used. In order to get only the teachers, the capability argument was provided to the function to narrow down the list of returned users to avoid showing other students in the course as supervisor candidates.

File API

In order to allow the user to download submitted files the file URL had to be generated via File API and then shown to the user. Once a user clicks on the file it will get downloaded to their machine.

Form API

Most of the views are generated using the Moodle built-in Form API with some additions of custom elements such as HTML tables and a couple of HTML forms with custom PHP submission logic.

The following Form API functions were used:

createElement - Used for creating new elements such as check-boxes, text fields, drop-downs, etc
addElement - Used for adding an element to the form
setType - Used for cleaning the input data from HTML tags
addRule - Used for making certain fields required and setting an error message for them
addHelpButton - Used for adding help text for some of the buttons
set_data - Used for setting existing information to the form. Example: opening a submitted thesis view sets the existing thesis data into the form
get_data - Used for getting the submitted data from the form
is_cancelled - Used for running functions and/or redirecting the user if the form was cancelled from
display - Action to display the form after having it defined

reset - Action that resets the form from any inserted data
addGroup - Used for adding custom form submission buttons such as confirming the thesis data, giving editing rights back to the student, providing feedback, etc
closeHeaderBefore - Used to close the header that was defined before group of elements
setDefault - Used to set a default value for an input field
hideif - Used to hide field(s) if another field has a certain value. For example hiding matriculation number input from thesis form if user is not marked as an author
disabledIf - Disable a field from being editable. Used to prevent the possibility of having 2 primary supervisors

The following **Form API** elements were used:

Header - Used to create a header component which can either extend or hide the components inside of it
Text - Used to create an input field of type text
Select - Used to create a select input field with a provided list of options
Hidden - Used to create a hidden component with a value which would be still saved into the form with an argument, passed either from the existing data or from URL parameters
Static - Used to create a static form element in order to display data that cannot be edited
Textarea - Used the same way as text input field, but has an easy way to customize input area size
Autocomplete - Used as a select input with the possibility of text search

Access API

The developed plugin is built on capabilities which represent specific functionalities that are available in the plugin. Capabilities can be matched with different roles, which allows customization of the process of sending theses from Moodle to Digikogu. This allows each institute or school to customize the thesis submission process to how they see fit.

Capabilities defined for this plugin are as follows:

1. **Add instance:** User with this capability is allowed to add the plugin to the course. By default only the administrator has the capability. Administrator also has the permission to change which roles can use this capability. If the capability is not applied to a role, the role cannot add an instance of the plugin to a course. It would

be recommended for an administrator to create an additional role in the system context [22] with this capability to allow specific users to add plugin instances to their courses. The latter solution can be used to avoid administrators having to add a new instance to a course every time it is asked for by study programs. The capability should not be given to mainstream roles as for example manager or teacher, as this would mean that anyone with the role can add a plugin instance to their course and as a result access Digikogu.

2. **View:** User with this capability can see and access the plugin instance on the course. If the capability is not applied to a role, the users with that role cannot access the instances of the plugin. By default student, teacher, non-editing teacher and manager roles have the capability.
3. **View all:** User with this capability can view all of the theses that are created in the current instance. Without this capability users might be able to see the plugin, the theses that they have created or they are supervisors of depending on other capabilities, however, they cannot see all the theses created in the instance of the plugin. By default the non-editing teacher and manager have the permission.
4. **View own students:** User with this capability can only view the theses that they are supervisors of. Users can become supervisors only if their role has been provided with the supervision confirmation capability. Without this capability users might be able to see all the theses of the instance of the plugin or they might see only their own thesis, however if they do not have the capability to view all theses, they will not be able to see the theses that they are supervisors of. By default the teacher role has the permission.
5. **Author confirmation:** User with this capability can increase the confirmation level of a thesis from the level 0 (author confirmation level) to the next level (1 - First confirmation level, 2 - second confirmation level or 3 - final confirmation level) depending on which confirmation and viewing capabilities are given to roles and in which stage a person with the capabilities to see and confirm the thesis in hand exists. This capability is also needed to be able to use the Update thesis data capability. If a person does not have this capability, they will not be able to confirm theses with confirmation level 0 and if they have the capability to update thesis data, they will not be able to do it as they don't have the permission to confirm the theses with confirmation level 0. Having no users with the author confirmation level for a thesis and the possibility to view the data means also that the process regarding that thesis cannot move forward and will be stuck. By default the student role has the permission.
6. **First confirmation:** User with this capability can increase the confirmation level of a thesis from the level 1 (First confirmation level) to the next level (2 - second confirmation level or 3 - final confirmation level) depending on which confirmation

and viewing capabilities are given to roles and in which stage a person with the capabilities to see and con rm the thesis in hand exists. This capability is also needed to be able to use the Update thesis data capability and the Re-enable author con rmation capability. If no user has the capability to provide rst con rmation for a thesis, the con rmation level will have been moved from 0 to 2 or 3 and no user can provide the rst con rmation. The process will move on with either second or nal con rmation capability. If a user has not been allocated the rst con rmation capability, the user cannot provide rst con rmations for the theses they can see, this also means that they cannot use other capabilities in that stage that are dependent on the con rmation capability. By default the teacher role has the permission.

7. Second con rmation: User with this capability can increase the con rmation level of a thesis from the level 2 (Second con rmation level) to the level 3 - nal con rmation level. This capability is also needed to be able to use the Update thesis data capability and the Re-enable author con rmation capability. If no user has the capability to provide second con rmation for a thesis, the con rmation level will have been moved from 0 or 1 to 3 and no user can provide the second con rmation. The process will move on with nal con rmation capability. If a user has not been allocated the second con rmation capability, the user cannot provide second con rmations for the theses they can see, this also means that they cannot use other capabilities in that stage that are dependent on the con rmation capability. By default the non-editing teacher role has the permission.
8. Final con rmation: User with this capability can change con rmation level of a thesis from 3 to level 4 (Ready for Digikogu level). This capability is also needed to be able to use the Update thesis data capability and the Re-enable author con rmation capability. If no user has the capability to provide nal con rmation for a thesis, the con rmation level will stay the same (0 or 1 or 2) as it is important that nal reviewing of the data is done. This way a user with the permission to change the capabilities of the roles can add the Final con rmation capability to a role and the process can move on. If there is no user that has the capability to provide nal con rmation, the process is stuck until xed as explained. If a user has not been allocated the nal con rmation capability, the user cannot provide nal con rmations for the theses they can see, this also means that they cannot use other capabilities in that stage that are dependent on the con rmation capability. Once a nal con rmation has been provided it is possible for the person with the nal con rmation capability to revoke the con rmation as long as a thesis has not been sent to the Digikogu. By default the manager role has the permission.
9. Update thesis data: User with this capability can edit theses that they have the viewing and one of the con rmation capabilities for. Edits can only be made when the con rmation capability is active. When the user does not have this capability,

they cannot make any changes to the theses that they can see and con rm. By default the student, teacher, non-editing teacher and manager roles have the permission.

10. Re-enable author con rmation: User with this capability can return the con r-mation permission back to the authors of the thesis. After an author re-con rms the thesis, the con rmation permission will return to the con rmation level of the user that gave the permission back to authors. This means that the whole process of con rmations will not be followed again. Con rmation permission can only be returned to author when the con rmation capability is active for the user. If a user does not have the capability, they cannot allow the author to make additional changes to their thesis even if the user has an active con rmation capability. By default the student, teacher, non-editing teacher and manager roles have the permission.
11. Submit: User with this capability can submit and make edits to their thesis if no theses are connected to the author within the plugin instance. If a user does not have this capability, they cannot create a new thesis by themselves. By default the student role has the permission.
12. Give feedback: User with this capability can provide feedback for the theses that they have the permission to view. If a user does not have this capability, they cannot leave any comments or feedback to theses they can see. By default the student, teacher, non-editing teacher and manager have the permission.
13. Insert defence date:User with this capability can insert defence dates to the theses that they have a permission to view. If a person does not have this capability, they cannot enter defence dates to theses they can see. If there is no user that can see a thesis and also insert a defence date for the thesis, the process of that thesis will be stuck as there is no-one that is able to insert a defence date and it is required to have a defence date connected to a thesis. By default the manager role has the permission.
14. Insert access restriction: User with this capability can insert access restrictions to the theses that they have a permission to view. If a person does not have this capability, they cannot enter access restrictions to theses they can see. If there is no user that can see a thesis and also enter an access restriction for that thesis, then no access restriction can be added to the thesis. By default the manager role has the permission.
15. Send thesis to Digikogu:User with this capability can send theses with the con-rmation level 4 (nal con rmation has been provided) to Digikogu if they have the permission to view the theses. If there is no user that can see a thesis and also send the thesis to Digikogu, the process will be stuck as there is no-one that is able to send the thesis to Digikogu. If a user does not have the capability to send thesis to Digikogu, they cannot send theses that they can see to Digikogu. By default the manager role has the permission.
16. Supervision con rmation: User with this capability can be selected as a supervisor

for a thesis. After that user is selected as a supervisor the user can con rm supervision of the thesis. Author cannot con rm their thesis if the supervisor selected has not con rmed supervision. Author con rmation can only be done when the supervisor con rms supervision or if the author changes supervisors data in their form and as a result either supervisor con rmation is added by another supervisor or data is inserted by hand and supervision con rmation is not needed. If no user has this capability, no users can be selected as supervisors in the thesis form and no user can con rm supervision. By default the teacher role has the permission.

The following capability combinations 1 per user role are recommended:

Capability	Author	Supervisor	Evaluation Committee	Digital Collector
Administrative capabilities				
Add instance	Not allowed	Not allowed	Not allowed	Optional
Viewing capabilities				
View	Required	Optional	Optional	Required
View all	Not recommended	Optional	Optional	Required
View own students	Not allowed	Optional	Not allowed	Not allowed
Con rmation capabilities				
Author con rmation	Required	Not allowed	Not allowed	Not allowed
First con rmation	Not allowed	Optional	Optional	Not allowed
Second con rmation	Not allowed	Optional	Optional	Not allowed
Final con rmation	Not allowed	Not allowed	Not allowed	Required
Capabilities dependent on con rmation capabilities				
Update thesis data	Required	Optional	Optional	Required
Re-enable author con rmation	Not allowed	Optional	Optional	Optional
Submit	Required	Not allowed	Not allowed	Not allowed
Other independent capabilities				
Give feedback	Optional	Optional	Optional	Optional
Insert defence date	Not allowed	Optional	Optional	Required
Insert access restriction	Not allowed	Optional	Optional	Required
Send thesis to Digikogu	Not allowed	Not allowed	Not allowed	Required
Supervision con rmation	Not allowed	Optional	Not allowed	Not allowed

Table 1.Allocation of capabilities

1. Student (Author)

   Add instance - must not be allowed due to security reasons

   View - must be allowed so that student can access the plugin and see their own thesis

   View all - should not be allowed for students to view theses that are not their own.

   View own students - should not be allowed because students should not be marked as supervisors.

   Author confirmation - must be allowed so that student can confirm their thesis submission. Otherwise process cannot continue.

   First confirmation - must not be allowed to avoid unchecked information reaching Digikogu

   Second confirmation - must not be allowed to avoid unchecked information reaching Digikogu

   Final confirmation - must not be allowed to avoid unchecked information reaching Digikogu

   Update thesis data - must be allowed so that authors can make changes to the data and files when a thesis has the author confirmation level and they have the right to see it.

   Re-enable author confirmation - must not be allowed to avoid unchecked information reaching Digikogu

   Submit - must be allowed so that student can submit a new thesis if no theses with the user as author exist in the plugin instance.

   Give feedback - is recommended to be allowed so that students can leave comments under the theses they can see or for example reply to the feedback that is left by other parties to their thesis.

   Insert defence date - must not be allowed as it is not part of authors' responsibilities to add defence dates and could cause wrong information reaching Digikogu.

   Insert access restriction - must not be allowed as it is not part of authors' responsibilities to add access restrictions and could cause wrong information reaching Digikogu.

   Send thesis to Digikogu - must not be allowed to avoid unchecked information reaching Digikogu.

   Supervision confirmation - must not be allowed to avoid authors becoming supervisors.

2. Teacher (Supervisor)

   Add instance - must not be allowed due to security reasons

   View - must be allowed so that teachers can access the plugin. If supervisors

are not meant to be engaged in the process this capability may not be applied but that means that no-one representing the party will be able to access the plugin.
View all - should not be allowed so that supervisors don't see the theses which are not supervised by them. If the process expects supervisors to see and also depending on other capabilities (con rmations, updating, re-enabling authors, etc) applied to them, to apply those functionalities on other theses than the ones that they are supervisors of, this capability can be applied.
View own students- must be allowed because supervisors must be able to see their students' submissions. Can be not applied if supervisors have the View all capability or supervisors are not engaged in the process and therefore, there is no need for them to be able to see the data and les of the thesis they supervise.
Author con rmation - should not be allowed so that supervisors would not be able to con rm the theses un nished by authors.
First con rmation - can be allowed to engage the supervisors in the review-ing process as the rst reviewers after an author has con rmed a thesis. If supervisors are wanted to be engaged in the reviewing process, only one of the con rmation capabilities should be applied. It is recommended to engage supervisors with the rst con rmation capability and evaluation committee with the second con rmation capability.
Second con rmation - can be allowed to engage the supervisors in the re-viewing process as the second reviewers after an author and possibly the rst con rmation have been provided. If supervisors are wanted to be engaged in the reviewing process, only one of the con rmation capabilities should be applied. It is recommended to engage supervisors with the rst con rmation capability and evaluation committee with the second con rmation capability.
Final con rmation - must not be allowed to avoid unchecked information reaching Digikogu.
Update thesis data- should be allowed if supervisors are engaged in the reviewing process and are expected to make changes to the data and les. The capability is active only when the user also has an active con rmation capability. If supervisors are not expected to change data or to be engaged in the reviewing process, this capability should not be applied.
Re-enable author con rmation - should be allowed if supervisors are engaged in the reviewing process and are expected to allow authors to make changes to the data and les if there is need for that. The capability is active only when the user also has an active con rmation capability. If supervisors are not expected to allow authors to make additional changes to the data and les of their thesis or to be engaged in the reviewing process, this capability should not be applied.

Submit - should not be allowed since supervisors should not be able to add submissions.

Give feedback- is recommended to be allowed so that supervisors can leave comments under the theses they can see or for example reply to the feedback that is left by other parties to theses. If supervisors are not expected to be able to leave feedback or be engaged in the reviewing process, this capability should not be applied.

Insert defence date- should not be allowed as it is not part of supervisors' responsibilities to add defence dates and could cause wrong information reaching Digikogu. Nevertheless, if the process should need supervisors inserting defence dates, the capability can be applied.

Insert access restriction- should not be allowed as it is not part of supervisors' responsibilities to add access restrictions and could cause wrong information reaching Digikogu. Nevertheless, if the process should need supervisors inserting access restrictions, the capability can be applied.

Send thesis to Digikogu- must not be allowed to avoid unchecked information reaching Digikogu.

Supervision con rmation - should be applied if supervisors are engaged in the process, if authors should be able to choose Moodle users with a speci c role as their supervisors and if supervisors should be able to con rm supervision of the theses. This capability is the only way to de ne the Moodle users as supervisors of speci c theses. If this capability is not applied, there will be no supervision con rmation process included.

3. Non-editing teacher (Evaluation Committee)

Add instance - must not be allowed due to security reasons

View - must be allowed so that non-editing teachers can access the plugin. If evaluation committee members are not meant to be engaged in the process this capability may not be applied but that means that no-one representing the party will be able to access the plugin.

View all - must be allowed so that non-editing teachers can see the theses created in the plugin instance. If evaluation committee members are not meant to be engaged in the process this capability may not be applied but that means that no-one representing the party will not be able to see all the theses in the plugin instance.

View own students- should not be allowed since it is covered by 'View all' capability. This capability is meant for supervisors only.

Author con rmation - should not be allowed so that a committee would not be able to con rm the theses un nished by authors.

First con rmation - can be allowed to engage the evaluation committee in the

reviewing process as the first reviewers after an author has confirmed a thesis. If evaluation committee is wanted to be engaged in the reviewing process, only one of the confirmation capabilities should be applied. It is recommended to engage supervisors with the first confirmation capability and evaluation committee with the second confirmation capability.

Second confirmation - can be allowed to engage the evaluation committee in the reviewing process as the second reviewers after an author and possibly the first confirmation have been provided. If evaluation committee is wanted to be engaged in the reviewing process, only one of the confirmation capabilities should be applied. It is recommended to engage supervisors with the first confirmation capability and evaluation committee with the second confirmation capability.

Final confirmation - must not be allowed to avoid unchecked information reaching Digikogu.

Update thesis data - should be allowed if evaluation committee is engaged in the reviewing process and is expected to make changes to the data and files. The capability is active only when the user also has an active confirmation capability. If evaluation committee is not expected to change data or to be engaged in the reviewing process, this capability should not be applied.

Re-enable author confirmation - should be allowed if evaluation committee is engaged in the reviewing process and is expected to allow authors to make changes to the data and files if there is need for that. The capability is active only when the user also has an active confirmation capability. If evaluation committee is not expected to allow authors to make additional changes to the data and files of their thesis or to be engaged in the reviewing process, this capability should not be applied.

Submit - should not be allowed since evaluation committee is not supposed to submit theses

Give feedback - is recommended to be allowed so that evaluation committee can leave comments under the theses they can see or for example reply to the feedback that is left by other parties to theses. If evaluation committee is not expected to be able to leave feedback or be engaged in the reviewing process, this capability should not be applied.

Insert defence date - should not be allowed as it is not part of evaluation committee's responsibilities to add defence dates and could cause wrong information reaching Digikogu. Nevertheless, if the process should need evaluation committee inserting defence dates, the capability can be applied.

Insert access restriction - should not be allowed as it is not part of evaluation committee's responsibilities to add access restrictions and could cause wrong

information reaching Digikogu. Nevertheless, if the process should need supervisors inserting access restrictions, the capability can be applied.
Send thesis to Digikogu - must not be allowed to avoid unchecked information reaching Digikogu.
Supervision con rmation - must not be allowed to avoid evaluation committee members becoming supervisors. If a member of the evaluation committee is also a supervisor, the user can be applied multiple roles and as a result they can use the capabilities of both roles.

4. Manager (Digital Collector)

Add instance - should not be allowed due to security reasons. digital collectors can be provided with an additional role that is allowed to use this capability. Mainstream role as for example manager should not be allowed to use this capability to avoid other users with the role adding the plugin instance to a course that is not meant for graduation theses.
View - must be allowed so that digital collector can access the plugin
View all - must be allowed so that digital collectors can see all the theses in the plugin instance and review submitted theses by either pushing them back to students to edit, by editing theses themselves or by sending theses to Digikogu
View own students - should not be allowed since it is covered by 'View all' capability. This capability is meant for supervisors only.
Author con rmation - should not be allowed so that digital collectors would not be able to con rm the theses un nished by authors.
First con rmation - should not be allowed so that digital collectors don't con rm un nished theses
Second con rmation - should not be allowed so that digital collectors don't con rm un nished theses
Final con rmation - must be allowed to enable digital collectors to provide nal con rmation for theses. Only with a nal con rmation the theses can be sent to Digikogu.
Update thesis data - must be allowed so that digital collectors can make nal changes to the data and les when a thesis has the nal con rmation level and a digital collector has the right to see the thesis.
Re-enable author con rmation - should be allowed so that digital collectors can send theses back to authors for some nal changes before providing the nal con rmation. The digital collector must have an active con rmation capability and the right to see the thesis to apply the functionality.
Submit - should not be allowed since digital collectors are not supposed to submit theses
Give feedback - is recommended to be allowed so that digital collectors can

leave comments under the theses they can see or for example reply to the feedback that is left by other parties to the thesis.
Insert defence date - must be allowed as it is required to have a defence date added to every thesis. Digital collectors must make sure that the defence date is added.
Insert access restriction - must be allowed as digital collectors must make sure that if a thesis has been applied an access restriction, the information would also be sent to Digikogu.
Send thesis to Digikogu - must be allowed as it is the key functionality that digital collectors must ful l - sending theses with correct data and les to Digikogu.
Supervision con rmation - must not be allowed to avoid digital collectors becoming Supervisors.

There are 4 con rmation levels that are also connected to the con rmation permissions. When an author con rms their thesis, the con rmation level changes from 0 to the next available con rmer's highest level. If a person who has the First con rmation right exists, the level changes to 1. Now the person with First con rmation permission con rms, con rmation level goes up again. If for example now a person with the Second con rmation right does not exist the con rmation level will change to 3 not 2. It is time for the person with Final con rmation permission to con rm the thesis. Once con rmed, the level changes to 4 which also means that the thesis is ready to be sent to Digikogu.

String API

String API is heavily used in the plugin's code in order to enable translation for texts, visible to the user. In order to enable the translations, a translation le was created in the lang/en folder and populated with values that follow the required format [18].

In order to use the aforementioned translations, each text value, shown to the user, must be de ned by calling get_string function with specifying what string should be read and the translation le name must be provided as the second argument.

At rst only English translations were added. When there were no new translation strings getting added, a new et folder was created in the lang folder and later populated with translations for the same le located in the en folder.

4.1.3 Digikogu partner API

First to be able to use the Digikogu Partner API at all, at the administrator settings of the plugin, the url and api-key must be con gured.

Figure 3. Digikogu Partner API Flow Chart
[19]

The project uses all of the stages of the Digikogu Partner API that are also explained in

the owchart of the Digikogu Partner API 3. When submitting a thesis to Digikogu from Moodle, rst a folder is created by using corresponding API request and the matriculation number of one of the authors. This also means that with every single matriculation number only one theses folder can be created.

Secondly, folders for les area created. The amount of folders depends on whether the theses has been applied an access restriction or not. If yes, then two folders are created - one with restriction and one without a restriction. If a thesis does not have an access restriction, only a folder without a restriction is created.

The third stage involves les being sent to the folders created. If a thesis has an access restriction, the summary le submitted by authors is sent to the folder without an access restriction and all the other les are sent to the folder with access restriction. If the thesis does not have an access restriction, the summary le is not sent at all and the rest of the les are sent to the folder without access restriction.

Fourth stage is for submitting the theses and if everything is correct the Graduation thesis plugin will receive a successful response.

Fifth stage is for making sure that the thesis exists in Digikogu. If the response to that is also positive, the user of Moodle will be presented a noti cation with a message stating that the thesis was sent to Digikogu successfully.

The `mod__graduationthesis __json` table contains every single request made to the API and all the responses received to keep a log for developers.

The table name `mod__graduationthesis __api` saves all the id-s received from the requests to be able to make following requests by using the id-s. Furthermore, knowing the id-s connected to matriculation numbers it is possible to access the thesis folders and if given thesis are not published also update the data in Digikogu.

4.1.4 Validation

In order to ensure that data is always correct, certain validation rules were added. Keywords are validated via regex, requiring the keywords to be listed in a comma separated list. All the e-mail inputs were validated via regex. When a thesis is receiving an author or rst level con rmations, the following thesis data is validated:

1. Authors have matriculation numbers de ned. Matriculation numbers are numeric

and are exactly 6 characters long
2. Thesis language is selected
3. Thesis has a title
4. There is a primary supervisor assigned to the thesis
5. All the supervisors, assigned to the thesis, have con rmed their supervision
6. Thesis must have keywords speci ed in Estonian and English languages
7. Thesis has a main le attached to it
8. Thesis has a summary le attached to it
9. Main and summary les are in PDF format

When giving nal con rmation, thesis must have a defense date speci ed.

During data submission, all the elds are also checked for having a positive length. This includes checking if all the author(s) and supervisor(s) data is entered.

When a person is providing their feedback, the feedback text must be provided. Cannot submit empty feedback.

4.2 Frontend

Most of the front end was developed using the Moodle built-in Form API in order to draw the main forms. The main data table, displayed on the module's landing page named `view.php` is implemented using HTML table with JavaScript functions embedded inside of it. In order to develop the ltering section of the page, a PHP form with POST method was added. The style of the form was described via CSS classes in order to make it more user friendly.

The page for con guring the added module is located in the module root directory and is named `mod_form.php`. Form for data input is generated using the Moodle Form API by adding each input eld to the form and setting required elds.

The page for viewing each thesis separately is located in the module root directory and is named `submitted_thesis_view.php`.

The page for adding/editing thesis information is located in the module root directory and is named `thesis_view.php`.

User interface design

User interface design is fairly simplistic and was implemented to require as few clicks as possible. Due to speci cs of the work ow and it relying on the Moodle capabilities, the user interface differs based on the user, their role in the course and the provided capabilities. Each page will be described in a more general way and differences with provided capabilities would be described separately.

Main view, which is de ned within view.php le, is implemented as a table and serves as an overview for the submitted theses and their statuses. The amount of shown theses and the available functionalities vary based on the capabilities given to the user. It was important to keep things in the same place to make it more accessible and easy for students, digital collectors and supervisors to use.

The most basic version of the main view is shown in the Figure 4 and illustrates how a person, who is within a group, belonging to the thesis grouping, sees the page. It shows a table with just one row, which shows the most important information about the thesis: status, author(s) and their matriculation number(s), supervisor(s), thesis title in Estonian and English, the main thesis document, defense date, indicator if an access restriction was provided for the thesis and for how long, whether or not a supervisor has con rmed that they are supervising the thesis and lastly the button to view the full thesis information.

Figure 4. Main view with a freshly created thesis

If a user is given an additional viewing capability, he would additionally see a ltering option. Theses can be ltered by author rst and last names, supervisor rst and last names, defense date and access restriction. This allows the user to quickly search a desired thesis among countless others. Along with ltering user can also sort the theses by order of creation, by Estonian or English titles, language, defense date, con rmation level, author rst and last names, their matriculation number as well as supervisor rst and last names. Both ltering and sorting happen in the back end via SQL query. For visuals see Figure 5.

If a user is marked as a thesis supervisor and has not yet con rmed his supervision, he would have an additional table, showcasing the submitted theses and a selection if they want to approve supervision. This offers a way of approving supervision and is much simpler than having to search through the existing theses to nd pending approvals. If a

Figure 5. Main view with the ltering option

supervisor has con rmed his supervision, he will also have a quick overview of theses he supervises. For visuals see Figure 6.

Figure 6. Supervision con rmation

If a user is allowed to either send the thesis to Digikogu, assign a defense date or set an access restriction period, an additional layer of buttons will be displayed under the theses table. Having these functionalities available in the same place allows the digital collectors to send theses to Digikogu in bulk with ease by selecting multiple theses and pressing the respective button. For visuals see the gure 7.

In order to view a single thesis, the user can click on the zoom icon under the Details header of the table. When clicking on it, user will be redirected to

Figure 7. Main view with extra button layer

`submitted_thesis_view.php`, which shows all the chosen thesis details. Figure 8 illustrates the basic view with the main sections collapsed.

Figure 8. Thesis collapsed overview

An additional layer of buttons that allow actions to be performed on the thesis is displayed in Figure 9

Figure 9. Thesis view action buttons

Author details gives a quick overview of the thesis author(s) information, including their first and last name(s), e-mail(s), matriculation number(s) as well as e-mail(s). The latter can be seen in Figure 10.

Supervisor details section displays the information about the supervisors of the thesis. The information includes supervisor's first and last name as well as their e-mail, see Figure 11.

The main information about the thesis itself that will be sent to Digikogu in addition to authors and supervisors data, is displayed in the Thesis details for Digikogu section, see Figure 12. Most of the details are specified in Estonian and English languages, thus will be described as one in the explanation below:

Figure 10. Author(s) section overview

Figure 11. Supervisor(s) section overview

EST/ENG title - Thesis title in Estonian and English languages
Language- Language which the diploma is written in
Estonian/English keywords- keywords, speci ed by author, in Estonian and English languages
Institution EST/ENG - Institution to which the thesis belongs to
School EST/ENG- School of the university to which the thesis belongs to
Thesis type EST/ENG- Type of the thesis. Can either be Masters, Bachelor or Graduation thesis
Study programme EST/ENG - The study program which the thesis is written in
Thesis document- Main thesis le, which can be downloaded if clicked
Summary document- Summary le, which can be downloaded if clicked
Appendices- Appendices, added by the author. Can be downloaded if clicked

The administrative information about the thesis is displayed in the Administrative thesis details section, see Figure 13. Details are explained below:

Ready for Digikogu - yes or no depending on whether the thesis has been provided with a nal con rmation and is ready to be sent to Digikogu.
Sent to Digikogu - yes or no depending on whether the thesis has been sent to Digikogu.
A list of con rmations - These indicate who and when con rmed the validity of the submitted data

(a) Author con rmation - con rmation from the thesis author
(b) First con rmation - con rmation from the thesis supervisor
(c) Second con rmation - con rmation from a user with second con rmation right. Intended for either the evaluation committee or supervisors
(d) Final con rmation - con rmation from the digital collector

Submitted feedback- List of all the provided feedback for the thesis. Consists of person's rst and last names, followed by the date when the feedback was posted and the feedback body itself

The nal section also offers certain actions to be executed on the thesis if a user has capabilities to do so. The following actions are allowed:

1. Giving feedback for the thesis
2. Con rming thesis edits to pass the thesis to supervisors to con rm or digital collectors to submit thesis to Digikogu
3. Giving authors the right to edit their submission after they have con rmed it
4. Making changes to the submission
5. Going back to the main view

Lastly there is the thesis edit view, the main menu where the user can make edits to the thesis data, including changing authors' and supervisors' information. The form consists of 3 main sections Figure 14:

1. Author(s) details Figure15
2. Supervisor(s) details where authors can select up to two supervisors including supervisors, who are not yet enrolled to the course by adding the data manually and selecting the primary supervisor for the thesis Figure16
3. Thesis information where thesis title, language, keywords and thesis related les can be submitted Figure17

Figure 12. Submitted thesis view data section

Figure 13. Submitted thesis view administrative data section

Figure 14. Thesis form overview

Figure 15. Author(s) input

Figure 16. Supervisor(s) input

Figure 17. Thesis data input

5. Analysis

5.1 Process overview

The overall process of graduation theses is explained in a le written by Ülle Laur. [23] While the latter is a good input for understanding what the current process is and looks like, the process for this project had to be developed by the authors. The process of sending theses from Moodle to Digikogu has been developed by authors relying on the process of the Informatics study program while joining meetings with various stakeholders and nding further information about the processes in other study programs and schools to make sure that the project could be engaged in the processes of more than just the Informatics study program. Throughout the process of the project there were many decisions made and analysis conducted by the authors as the support and answers to questions from some of the clients were not as quick as needed for the project. [24], [25] Some of the key stages and elements of the project analysed by the authors are as follows.

5.1.1 Data types

The data that is asked to be lled for every single thesis is mainly connected to the requirements of the Digikogu Partner API. The exceptions are the emails of authors and supervisors. The administrative data is added to support the process and user experience. Seeing who and when has con rmed the thesis, whether a thesis is ready for Digikogu or is already sent to Digikogu aims to provide the users with information on in what stage the process is. Feedback is a supportive functionality that helps users to communicate to each other if needed. The administrative data was added to the project after realising that users need some support in understanding how the process of the plugin works.

5.1.2 Data insertion stages

As it was important to decrease the amount of mistakes done by entering data, the data that must be inserted was split into two - the data that is the same for all the theses in a study program (general data) and can be entered by only one person instead of every author inserting the same data and the data that must be de ned separately for each thesis. As a result, there is the plugin instance settings view that supports entering the institution,

school of the university, thesis type and study program. All this data will be added to every single thesis that is created in the plugin instance. The rest of the data is entered per each thesis.

General data

Authors rst developed a solution that supported creating multiple groups with different general data so that in one plugin instance it would be possible to have the theses of for example different study programs. Nevertheless, as the solution also included dynamic elds that would appear or disappear according to the needs of the process, many bugs started to appear as Moodle documentation does not have strong support for such cases. As a result, it was decided to keep the plugin simple meaning that each plugin instance represent only one type of general data. In one course multiple plugin instances can be added and each instance can hold different general data.

5.1.3 Data presentation

In the thesis view authors decided to split the data into four groups including the authors data, supervisors data, data that is needed for Digikogu and administrative data to make it easier for reviewers to nd information. As a result, the reviewers can notice right away where to nd the information they are looking for instead of having to read through every single line of data to nd what they are looking for.

5.1.4 Main view data

In the main view, where the data of multiple theses is presented, it was realised by the authors that only limited data elds should be covered there that would support the users in nding speci c theses. As a result, only the status, authors' and supervisors' details, title of the thesis, main thesis le, defence date and access restriction were included in the view.

5.1.5 Availability of functionalities

Key functionalities can be found both in the main view and the thesis view. It was decided to support adding defence dates, access restrictions and sending thesis to Digikogu in the main view, as these are some functionalities that the digital collectors can apply on multiple theses at the same time. In the thesis view the feedback, editing, giving the con rmation right back to the authors and con rming a thesis functionalities are presented, as these are the functionalities that should be used only after reviewing all the data of the thesis.

The editing functionality and giving con rmation right back to authors functionality are only available when the user also has an active con rmation capability. This decision was made as the theses should not be edited by other parties when it is going through a reviewing process of one party.

Functionality to provide feedback is accessible all the time to make sure that if any of the parties have any comments or problems with a thesis even if they do not have the con rmation right, they would be able to bring that information to the attention of the authors or reviewers. Furthermore, the possibility to provide feedback is also available once the theses have been sent to Digikogu to for example provide authors with the information that sending their thesis to Digikogu was successful and share a link to the address of their thesis.

The functionality of sending theses to Digikogu is available to anyone that has the capability at any time. This decision was made because the functionality applies on only the theses that have been provided with the nal con rmation and cannot be edited unless the nal con rmation is revoked, in which case the con rmation level is changed and person with the capability of sending theses to Digikogu cannot apply the functionality on the thesis anymore. Therefore, there is no risk connected to someone making changes to the thesis while it is being sent to Digikogu.

Revoking the nal con rmation functionality was made possible as it was realised that the nal con rmation could be provided by mistake and therefore, there is a need for being able to revoke the nal con rmation. Otherwise the process would get stuck if the nal con rmation is provided but a new mistake is noticed in the thesis and as a result is not ready for it to be sent to Digikogu.

5.1.6 Con rmations process

The con rmations process was developed to make sure that depending on the process of the study program, various stakeholders could be engaged in the process. Authors realised that the key parties engaged in the process are authors, supervisors, evaluation committee and digital collectors. While in the process of submitting and sending data and les to Digikogu the two parties that are essential to the process are authors and digital collectors, the author and nal con rmation capabilities were created. These are the two con rmation capabilities that must be included in the process in the plugin. The rst con rmation and second con rmation capability were created to support the processes that wish to engage additional parties as for example supervisors and evaluation committee in the reviewing process. Furthermore, it was realised that the more stakeholders review the data the less

mistakes will reach Digikogu.

Additional analysis was done by authors to nd a way for skipping con rmations when a thesis has no reviewer in a given stage.

It was decided by the authors that if a user has multiple con rmation capabilities then only one con rmation is provided at a time. This means that for example when a user has both the rst and second con rmation permission, then they must rst con rm the thesis with the rst con rmation and then either the same person or another user with the second con rmation capability can provide the con rmation.

Authors also decided that if the con rmation right is given back to authors, after an author re-con rms the thesis, the con rmation level will change to the level that the thesis had before the author received the right. This means that the con rmations that had already been provided by various reviewers will not have to review the theses again.

5.1.7 Supervision con rmation

It was decided by the authors to include a supervision con rmation stage to the process. First of all, it was made possible for authors to choose a user of Moodle as their supervisor instead of having to type in the details of a supervisor. If a supervisor is selected from the users list, the user must con rm supervision of the thesis. The latter was included to make sure that authors will not be able to choose random supervisors as their own. Before an author can provide their author con rmation to a thesis, selected supervisors must con rm supervision.

5.1.8 Views versus capabilities

At rst the authors started developing the project by creating separate views to each of the stakeholders. However, soon it was realised that rst of all, many of the elements of views are repeating and secondly there is no clear structure to the process and stakeholders. Therefore, information about capabilities was found and it was decided to build the process on capabilities rather than roles and their views. Capabilities provided the possibility to play around with the roles and functionalities presented in views.

5.1.9 Assignment plugin

It was a requirement by the university that Assignment plugin must be used for les submission, however it was not clear how it can be integrated with the project. As a result of some analysis, reading and looking into the database of Moodle, a way was found that makes the process simple and logical. Authors must submit their documents through one to three Assignment Plugin instances. The instances will be connected to the Graduation thesis plugin instance and while lling in the metadata, authors can select the les that they or their co-authors have submitted in the Assignment plugin instances.

5.1.10 Groups API

It was important that the plugin would also support group graduation theses as this is one of the possible processes within the School of Information Technologies. As the Group self-selection plugin, that is supported by the Groups API of Moodle, is already in use in some of the study programs, the possibilities to use the Groups API for creating theses groups and creating theses based on that was looked into. At rst, the authors tried to come up with a logic, where all the stakeholders would be included in a group and when the theses are created based on those groups, authors would have to de ne the roles of various stakeholders. [26] Soon, however, it was realised that the best way would be to only include authors, which excluded an additional and unnecessary step (role de ning) in the process. Rest of the stakeholders are de ned by capabilities.

5.2 Valuable meetings

During the project meetings with the key client Ago Luberg were hold every second week, where issues were discussed and decisions were made on how to move forward. Additionally, three valuable meetings took place, where various stakeholders were invited to provide the project with some feedback and discuss some of the key problems.

5.2.1 26th January: Meeting with all of the clients

On 26th January there was a meeting where the details of doing this project as a graduation thesis project were discussed. In addition to the representatives of this project, the meeting was attended by:

1. Ago Luberg, who is the supervisor of this project and the primary client of this project representing the School of Information Technologies.

2. Liisi Järve and Anna Varkki, who were representing the secondary client - Of ce of Academic Affairs of the university.
3. Katrin Valvik, who was representing the Information Technology Services department of the university.

As a result of this meeting it was agreed that the project will be further developed for the School of Information Technology with the possibility in mind that if it is a success, further schools or study programs can test the plugin. [27] After the meeting feedback for the prototype that was created during an internship in summer 2021, answers to the questions presented by an author of the thesis and requirements for the new project were provided by Katrin Valvik and Anna Varkki.

5.2.2 2nd February: Meeting with stakeholders

On 2nd February there was a meeting where the old version of the project was presented to digital collectors and their feedback was asked for to gain further understanding of their needs. In addition to the authors of this work and clients including Ago Luberg, Liisi Järve, Anna Varkki and Katrin Valvik, approximately 20 administrative employees of the university that are connected to the process of sending theses to Digikogu, that took part in this meeting. There were many questions discussed at the meeting and the overview of it is stored in Con uence. [28]

5.2.3 22nd April: Demo to clients and couple of digital collectors

On 22nd April the project was presented to the clients including Ago Luberg, Anna Varkki, Liisi Järve and Katrin Valvik but also to couple of digital collectors including Riina Soovik and Kairi Raspel. Additionally, a few employees of the Information Technology Services department joined the meeting. The aim of this meeting was to gain further insight to whether the stakeholders see value in the project and if the authors were moving in the right direction. As a result, Riina Soovik stated that she is looking forward to testing and using it.

6. Results and Validation

As a result of this project a Moodle activity plugin named Graduation thesis was created to allow digital collectors use the data inserted by students for sending theses straight to Digikogu instead digital collectors having to type the data in by themselves. The following functionalities were added to the plugin as capabilities that can be assigned to different roles of Moodle to allow personalisation of the plugin according to the needs of different schools of the university to organise their processes in Moodle as they wish.

1. Administrative capabilities
   (a) Permission to add plugin instances to a course (Add instance capability)
2. Viewing permissions
   (a) Permission to view only own theses and access a plugin instance (View capability)
   (b) Permission to view only theses under user's supervision (View own students capability)
   (c) Permission to view all theses in plugin (View all capability)
3. Con rmation permissions (further explanation provided below)
   (a) Author con rmation permission (Author con rmation capability)
   (b) First con rmation permission (First con rmation capability)
   (c) Second con rmation permission (Second con rmation capability)
   (d) Final con rmation permission (Final con rmation capability)
4. Permissions dependent on con rmation permissions
   (a) Permission to change the metadata and les of theses (Update thesis data capability)
   (b) Permission to give editing permission back to the author (Re-enable author con rmation capability)
   (c) Permission to create a new thesis (Submit capability)
5. Other independent permissions
   (a) Permission to give feedback (Give feedback capability)
   (b) Permission to insert defence date (Insert defence date capability)
   (c) Permission to insert access restriction (Insert access restriction capability)
   (d) Permission to con rm supervision (Supervision con rmation capability)
   (e) Permission to send theses to Digikogu (Send thesis to Digikogu capability)

The solution supports adding additional reviewers to the process of making sure the metadata and les inserted are correct. Graduation thesis plugin allows one platform to be used for data and les insertion, reviewing and submission to Digikogu instead of using multiple communication channels (exception is the study information system, which is part of the university's proccesses where the same data needs to be inserted and reviewed as well). Furthermore, in the project attention is paid to missing data and possible mistakes. As a result, the workload of digital collectors is decreased. The latter, however still needs to be validated outside of the scope of this project.

To validate the overall success of the project it must be tested on a graduation thesis course for a whole semester and this does not t the timescale of the project. However, the initial feedback from digital collectors, especially Riina Soovik - a digital collector of the School of Information Technologies after having shown them the result, is that the plugin is de nitely needed, would bene t their work ow and will be put to use once it is live on the Moodle of Tallinn University of Technology.

From the meetings and conversations with the Educational Technology Centre, Software Development Division, and digital collectors it was pointed out to:

include a few keywords to thesis' keywords automatically, including the theses type as for example bachelor thesis;
make it possible for digital collectors to assign access restrictions to thesis instead of students while students will have to provide thesis summary as a separate le just in case a restriction is applied.
set how many years after defence day the access restriction will last instead of setting a precise end date for the access restriction;
make it possible for digital collectors to change the data and les inserted;
present validation errors if possible right away in the table instead of presenting them in a pop-up alert window;
validate data before any actions are taken;
to avoid using questionnaire for data insertion
to make sure the log of Digikou API requests and responses is available for the developer

All of the points mentioned were taken into consideration and implemented.

7. Comments

As the support and decision making from the university side takes more time than the time-scale of this project allows, the project of sending graduation theses from Moodle to Digikogu has involved a lot of developer side analysis, process-description and testing out various possibilities that Moodle provides. Additionally, as the university did not provide the project with a clear structure and processes, important part of the project was also to come up with the design that would support the possible processes that the university will want to follow in the future, hence flexibility was kept in mind throughout the project, while still making sure that there is a clear system and logic to the plugin. Coming up with the following processes took some more time and remodelling by authors.

Using capabilities instead of specific roles: At the beginning the authors of this work started building views for each separate role. After some testing and analyzes of the process it was realized that it would benefit the project to build the project on capabilities that support a flexible solution where the structure of the process can be changed by the application of capabilities.

Integrating groups: Groups API was decided to be used first of all, to manage group theses but also, to include supervisors and other parties to the groups. Nevertheless, the latter seemed to cause more confusion and complexity in code than help with the process. Furthermore, as other parties were supposed to be assigned a role it created a possibility for additional mistakes that would only increase the workload of the employees of the university. As a result, it was decided to use groups only for grouping authors and not for other parties connected to the thesis. Supervisors are connected to the thesis in the submission form.

Not using one plugin instance for multiple theses types/study programs/departments: while at first a solution was created for allowing to create multiple theses types under one plugin instance, for example to have theses of two study programs under the same plugin instance, it was decided to remove the solution. First of all, Forms API in Moodle is not the friendliest when it comes to unsaved changes in the form where fields are dependent on each other and as a result, validation and prevention of mistakes turned out to be over-complicated and possibly impossible. Secondly, it was realised that the functionality is not needed as multiple plugin instances can be added to one course. The plugin instances can be created with

different institution, school, study program and thesis type data. Course would then have multiple Graduation thesis plugin instances with different theses in them. The latter is useful when for example one of the schools is interested in the Digital Collector managing all the theses under one course.

The prototype created during an internship in 2021 summer, provided the authors with great insight of what works and what does not. Some of the key ideas taken into account:

Questionnaire: during the internship it seemed a good idea to use the already existing Activity plugins for the processes of sending theses from Moodle to Digikogu. One of the plugins used was Questionnaire. In the prototype entering the metadata was built on the latter. Nevertheless, it turned out that is caused more issues than expected, furthermore it was nearly impossible to personalise it to the needs of the project as it is part of the Moodle core and therefore, the code should not be changed. In the current project questionnaire was not used and the form for inserting metadata was developed using the Form API.

Data views: it was noted that presenting all the theses with all the data on one page as a table was overwhelming. After consulting with the digital collectors it was decided to have two views for data - one in a table to have an overview of the the existing theses and the other view for only a selected thesis to see all the data connected to it.

8. Conclusion and Future Development

Digitalisation of universities is inevitable if the universities wish to be competitive at the market [29]. The latter also means integrating the technologies used at the university to support the processes of the universities and decreasing the workload of employees by creating a seamless integration between various systems used [30].

The integration between Moodle and Digikogu built as a result of the project in hand aims to support collaboration between students, teachers and the employees of the Tallinn University of Technology and as a result cut down the workload of digital collectors, accelerate the process of sending thesis to Digikogu and drop the amount of mistakes made during the process.

As a result of the project there was an activity plugin created that supports the key activities of the process:

students submitting their thesis data and les;
reviewers (supervisors, evaluation committee, digital collectors or other if needed) reexamine the data and les (make changes to data and les, give feedback, allow students to make additional changes and submit a con rmation of the reviewer)
stakeholders inserting defence dates and access restrictions
digital collectors sending thesis to Digikogu

In addition to the key activities, the plugin also provides a possibility to lter and sort thesis by various data, to make the process quicker for reviewers and validates the data to decrease the amount of mistakes made during the process.

Overall, initial solution was created that integrates the two heavily used platforms. Nevertheless, in the future the plugin needs to be tested in different schools of the university and further analysis must be conducted to understand the real bene t of the plugin. Users feedback should be collected to understand how it can be improved.

The plugin will very likely be uploaded to the Tallinn University of Technology Moodle platform for the School of Information Technologies to test it out during the upcoming fall

of 2022.

There are a few ideas that should be considered for future developments.

Noti cations

First, it would be bene cial to add noti cations functionality to the plugin. This means that when one of stakeholders provides a con rmation, the stakeholder that is in line to react to the theses should receive a noti cation with corresponding information. [31]

Changes log

Add an additional view to the plugin with the possibility to view all the changes made to a theses and the details of the editor. [32]

Browser view for les

Files are currently downloaded when clicking on the le in the plugin. It would create a better user experience if the les were opened in the browser instead of downloading them. [33]

Grading functionality

Add the possibility to add nal grades of the thesis to the plugin and connect it with the Gradebook API. [34]

Processor API

Connect the plugin with the Processor - the managing system of graduation thesis topics of the School of Information Technologies [35]. The Processor has an API built to it, which the graduationthesis plugin can use to request the data of the thesis including the title, authors and supervisors of the thesis.

Integration with the study information system

A key to really building a seamless integration for sending thesis to Digikogu is integrating Moodle and the study information system (ÕIS) of the university and making it possible to request the data submitted by students at ÕIS from Moodle. This would result in a process where students would not have to type in the data twice and digital collectors would not have to review the data twice (once in ÕIS and once in Moodle). To start off with this project, a Graduation thesis API could be created that would support data exchange with the study information system.

Add the possibility to include more than two supervisors

At the moment the project is built according to the Academic Policies of the Tallinn University of Technology, which states 'a graduation thesis may have up to two supervisors'. [36] Thus, only two supervisors can be added to a thesis. Nevertheless, rules as such can change in the future and it would bene t the project if administrators of Moodle could change the amount of supervisors that can be added to a thesis.

Dynamical con rmation capabilities

Currently there is a possibility provided to build a process consisting of four con rmation capabilities. While it is enough to include the authors, supervisors, evaluation committee and digital collectors to the reviewing process, it could bene t the project if there was a possibility to include in nite number of con rmations in case a need for that occurs. Therefore, the possibility to turn con rmation capabilities into a dynamic con rmation capability, where a number of con rmations needed for the process can be decided in the user interface, should be looked into.

Bibliography

[1] TALTECHDIGITAL. Accessed: 2022-05-01. 2021. URL: https://taltech.ee/en/taltechdigital.

[2] Tallinn University of Technology. I ja II astme õppeprogrammide üliõpilaste edasijõudmine. Tech. rep. Accessed: 2022-05-01. 2021. URL: https://portal.taltech.ee/v2/powerbi.

[3] Kimberly A. Barchard and Larry A. Pace. "Preventing human error: The impact of data entry methods on data accuracy and statistical results". In: Computers in Human Behavior 27.5 (2011). 2009 Fifth International Conference on Intelligent Computing, pp. 1834–1839. ISSN: 0747-5632. DOI: https://doi.org/10.1016/j.chb.2011.04.004. URL: https://www.sciencedirect.com/science/article/pii/S0747563211000707.

[4] Ülle Laur. Lõputööde avalikustamine Digikogus. Tech. rep. Accessed: 2022-05-01. 2021. URL: https://con uence.ttu.ee/pages/viewpage.action?pageId=43685019.

[5] What is Ouriginal. Accessed: 2022-05-01. 2021. URL: https://www.ouriginal.com/our-products/.

[6] Roles and capabilities. Accessed: 2022-04-29. 2020. URL: https://docs.moodle.org/400/en/Roles_and_permissions.

[7] Grouping users. Accessed: 2022-04-29. 2020. URL: https://docs.moodle.org/400/en/Grouping_users.

[8] R. Barras. Group self-selection. Accessed: 2022-04-29. 2020. URL: https://moodle.org/plugins/mod_groupselect.

[9] Activity module development for Moodle: a sample activity module, edugame. Accessed: 2022-05-01. 2009. URL: https://www.researchgate.net/publication/262327291_Activity_module_development_for_Moodle_a_sample_activity_module_edugame.

[10] Plugin types. Accessed: 2022-05-01. 2021. URL: https://docs.moodle.org/dev/Plugin_types.

[11] "Frontend vs Backend". In: (2021). Accessed: 2022-05-01. URL: https://geeksforgeeks.org/frontend-vs-backend/.

[12] Groups API. Accessed: 2022-04-30. 2021. URL: https://docs.moodle.org/dev/Groups_API.

[13] *Data manipulation API*. Accessed: 2022-05-01. 2022. URL: https://docs.moodle.org/dev/Data_manipulation_API.

[14] *Enrollment API*. Accessed: 2022-05-01. 2021. URL: https://docs.moodle.org/dev/Enrolment_API.

[15] *File API*. Accessed: 2022-05-01. 2022. URL: https://docs.moodle.org/dev/File_API.

[16] *Form API*. Accessed: 2022-05-01. 2022. URL: https://docs.moodle.org/dev/Form_API.

[17] *Access API*. Accessed: 2022-05-01. 2022. URL: https://docs.moodle.org/dev/Access_API.

[18] *String API*. Accessed: 2022-05-01. 2021. URL: https://docs.moodle.org/dev/String_API.

[19] *Digikogu Partner API*. Tech. rep. Accessed: 2022-05-01. 2021. URL: https://confluence.ttu.ee/display/MDL/Digikogu+API+kirjeldus.

[20] Paul Namuag. *Which Moodle Database Should You Use?* Accessed: 2022-04-30. 2021. URL: https://severalnines.com/database-blog/which-moodle-database-should-you-use.

[21] *PHP coding standards*. Accessed: 2022-05-01. URL: https://raw.githubusercontent.com/php/php-src/master/CODING_STANDARDS.md.

[22] *Assign roles*. Accessed: 2022-05-28. 2021. URL: https://docs.moodle.org/400/en/Assign_roles.

[23] Ülle Laur. *Protsessi analüüs*. Tech. rep. Accessed: 2022-05-29. 2021. URL: https://confluence.ttu.ee/x/m5SaAg.

[24] Anna Sooniste. *Moodle-Digikogu liidestamine ning lõputööde digikokku saatmise protsess*. Tech. rep. Accessed: 2022-05-29. 2022. URL: https://docs.google.com/document/d/16qhDzXWy-DzyitycAChb7LMuGSyM-bZHJP_Ma-0ntw8/edit?usp=sharing.

[25] Anna Sooniste. *Põhiprotsess ja funktsionaalsused*. Tech. rep. Accessed: 2022-05-29. 2022. URL: https://docs.google.com/document/d/1SraxnKTjXcPKOgYGnl10sMEFsIf-ljwYosT6dSf2CXI/edit?usp=sharing.

[26] Anna Sooniste. *Process*. Tech. rep. Accessed: 2022-05-29. 2022. URL: https://www.figma.com/file/7p19x9YoHlJ1uHIOvePKC8/Process?node-id=0%5C%3A1.

[27] Katrin Valvik. *2022-01-26 Lõputööde plugina edasiarenduste perspektiiv*. Tech. rep. Accessed: 2022-05-29. 2022. URL: https://confluence.ttu.ee/pages/viewpage.action?pageId=58170390.

[28] Katrin Valvik. *2022-02-02 Lõputööde avalikustamine digikogus (IT lahenduse arutelu)*. Tech. rep. Accessed: 2022-05-29. 2022. URL: https://confluence.ttu.ee/pages/viewpage.action?pageId=58170896.

[29] John Holmwood and Chaime Marcuello Servos. "Challenges to public universities: Digitalisation, commodification and precarity". In: *Social Epistemology* 33.4 (2019), pp. 309–320.

[30] Sirje Virkus et al. "Integration of digital libraries and virtual learning environments: a literature review". In: *New Library World* (2009).

[31] Anna Sooniste. *Send notification when there is a theses to confirm*. Accessed: 2022-05-01. 2022. URL: https://gitlab.cs.ttu.ee/ained/graduationthesis/-/issues/69.

[32] Anna Sooniste. *Create a log view for theses changes: who changed what and when. Requires a new db table?* Accessed: 2022-05-01. 2022. URL: https://gitlab.cs.ttu.ee/ained/graduationthesis/-/issues/93.

[33] Anna Sooniste. *Instead of downloading files present them in browser*. Accessed: 2022-05-01. 2022. URL: https://gitlab.cs.ttu.ee/ained/graduationthesis/-/issues/96.

[34] Anna Sooniste. *Add grading functionality*. Accessed: 2022-05-01. 2022. URL: https://gitlab.cs.ttu.ee/ained/graduationthesis/-/issues/102.

[35] Ruuben Risto. "Lõputööde teemade haldamise rakendus". MA thesis. Tallinn, Estonia: Tallinn University of Technology, 2020. URL: https://digikogu.taltech.ee/et/Item/ec74957c-49b4-4254-a1fa-87688a1f04d2.

[36] *Academic Policies*. Accessed: 2022-05-28. 2021. URL: https://oigusaktid.taltech.ee/en/academic-policies/.

Appendices

Appendix 1 - Database Diagram

Find the database diagram connected to Moodle core database tables HERE.

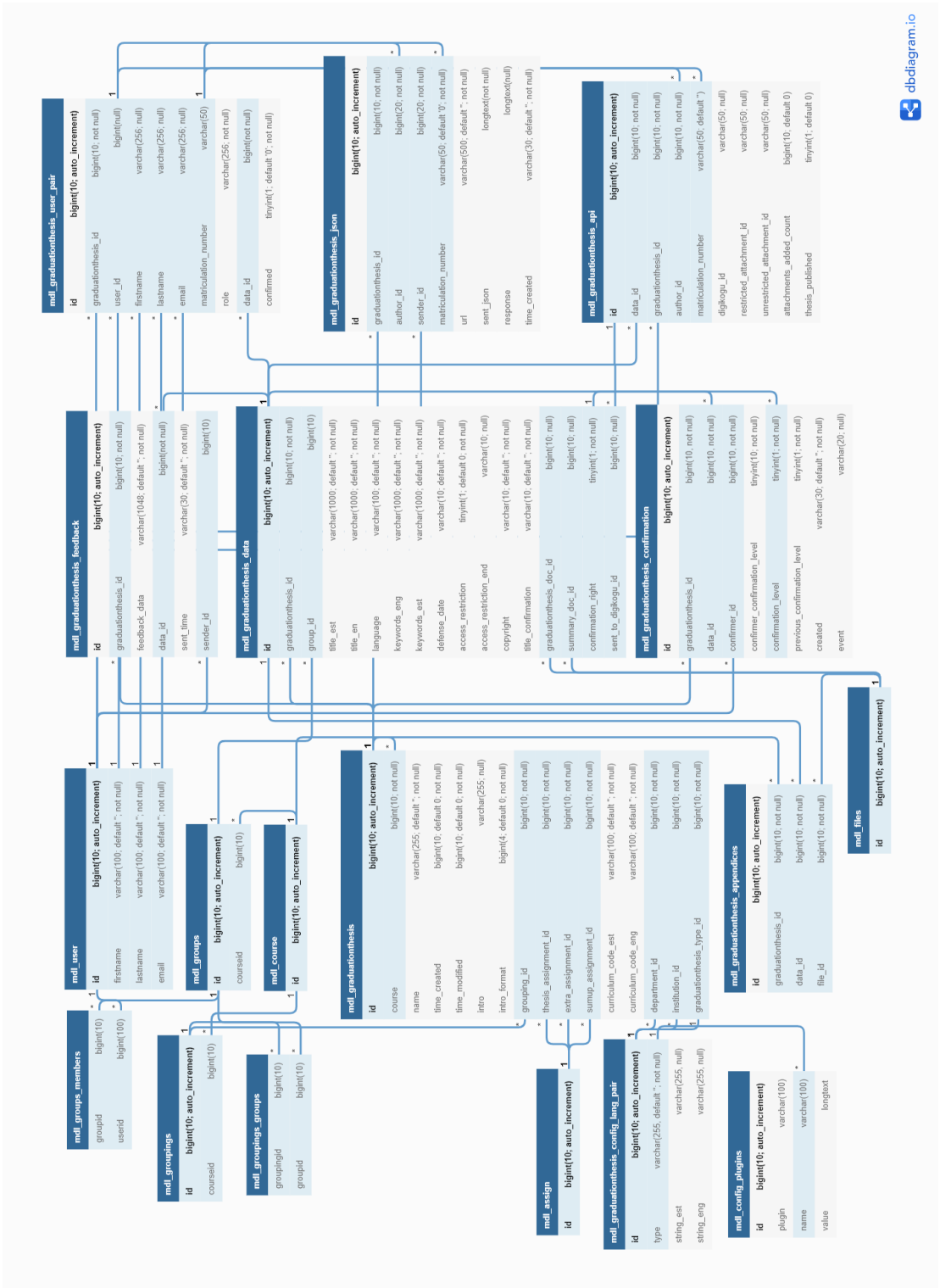

Figure 18. Database diagram highlighted

Find the database diagram with only the database tables that were added as part of the project HERE.

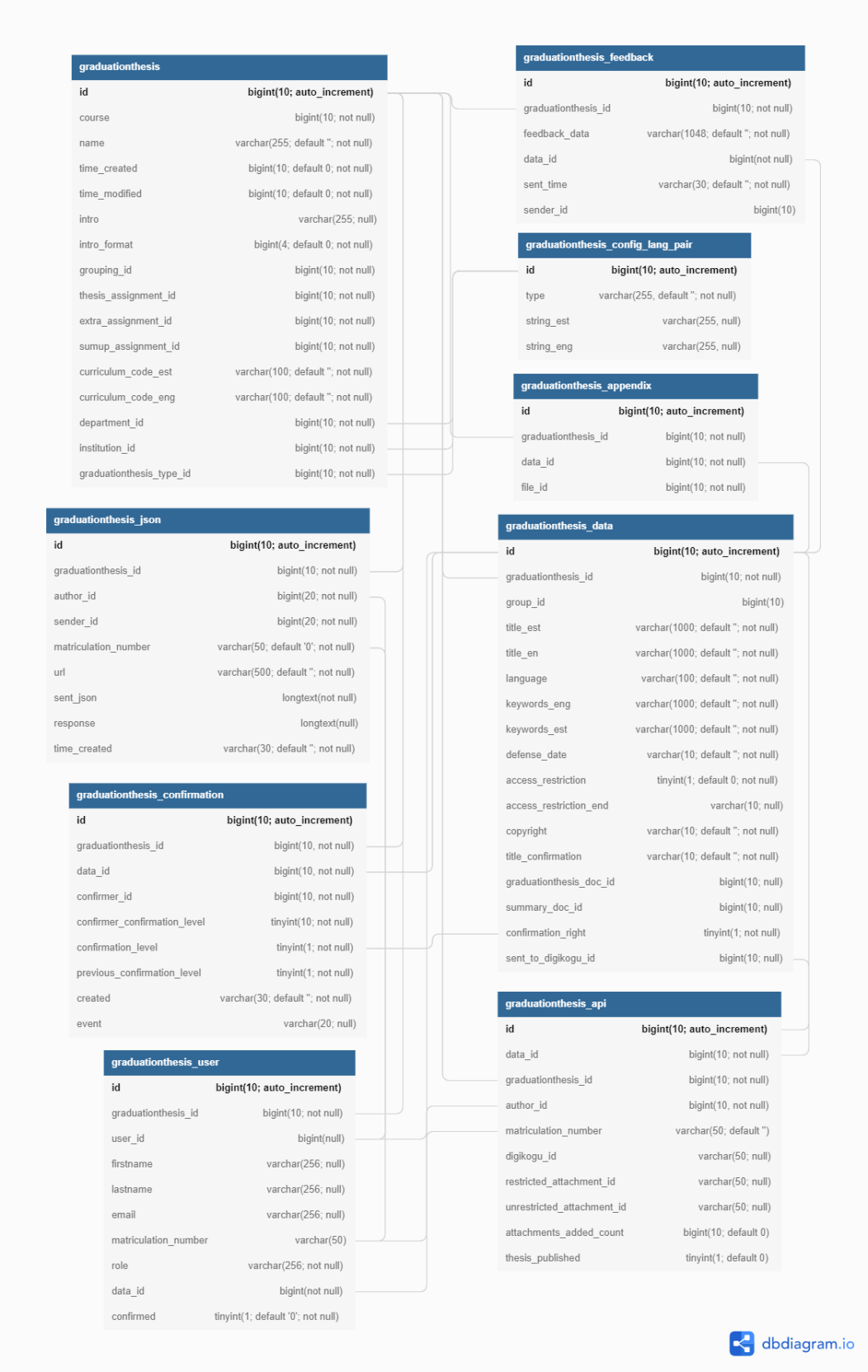

Figure 19. Project centered database diagram

Figure 20. Project centered database diagram highlighted

Appendix 2 - Rules for Publication and Preservation of Graduation Thesis

Approved by Rector's directive No 17 of 7 April 2020

In force from: 01.10.2020

Rules for Publication and Preservation of Graduation Thesis

The directive is issued based on subsection 23 (6) of the Academic Policies established by Regulation No 6 of 17 December 2018 of the Senate of Tallinn University of Technology.

1. General provisions

1.1 These Rules apply to publication and preservation of graduation thesis submitted for applying for professional higher education diplomas, bachelor's degrees and master's degrees (hereinafter graduation theses) at TalTech.

1.2 The non-exclusive licence (Annex 1 to the Rules) forms part of a student's graduation thesis. The non-exclusive license must be submitted in one file together with the electronic version of the thesis (in PDF format) and this is one of the prerequisites for allowing the author to defend his/her thesis and ensuring the protection of his/her copyright. The non-exclusive licence does not require a signature.

1.3 The author of a thesis is liable for ensuring that the publication of the graduation thesis does not infringe other persons' rights arising from specific legislation regulating the field of intellectual property, the Personal Data Protection Act, the Copyright Act and other legislation.

1.4 The author of a graduation thesis can submit an application for establishing restrictions on the publication of the thesis in cases prescribed by law.

2. Publication of graduation theses

2.1 The employee appointed in accordance with the school's internal procedures shall publish the graduation thesis that has no access restrictions and that has been given a positive grade, including the multimedia files (audio, visual or audiovisual recordings) that form part of the thesis, in the digital collection of the library immediately after the defence

of the thesis.

2.2 If it is not possible to publish the graduation thesis because the economic rights of the author belong to another person or the thesis contains personal data, state secret, classified information of foreign states or any other classified information, the student shall apply for restriction on access to the graduation thesis in compliance with the procedure established at the school.

2.3 If access to the graduation thesis is restricted, only the abstract of the graduation thesis is published during the validity period of the restriction.

2.4 The author of a graduation thesis with restricted access shall prepare an abstract, which shall appear after the front page of the title page. An abstract shall include the name of the author and the title of the graduation thesis. An abstract must not contain confidential aspects of the thesis.

2.5 A graduation thesis with restricted access shall be published in the digital collection of the library immediately after expiry of the restriction.

3. Preservation of graduation theses

3.1 TalTech ensures the preservation and availability of the electronic version of the graduation theses.

4. Implementing provisions

4.1 The Procedure for Publication and Preservation of Graduation Thesis approved by Rector's directive No 60 of 27.02.2014 is repealed.

4.2 This directive shall enter into force on 1 October 2020.

Annex to Rector's directive No 1-8/17 of 7 April 2020

Non-exclusive licence for reproduction and publication of a graduation thesis1

We, Anna Sooniste, Ruslan Nesterov and Ewert Ubaleht

1. grant Tallinn University of Technology free licence (non-exclusive licence) for our thesis Moodle and Digikogu Integration: Developing a Moodle Plugin for Sending Graduation

Theses to Digikogu, supervised by Ago Luberg.

1.1 to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;

1.2 to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.

2. We are aware that the authors also retain the rights specified in clause 1 of the non-exclusive licence.

3. We confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

30.05.2022

1 The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.