University of Twente

Design Report

Group 14

J. Bovenkerk
W. van den Brink
A. Koç
M. Meijer
D. Vos

Supervisor

M. Huisman

April 22, 2022

Abstract

For the *Design Project* module at the University of Twente, we have developed a system for the project "Leren Zichtbaar Maken" for Lyceum De Grundel in Hengelo. The system aims to create an atmosphere for students and teachers to visualize the learning process. Students can scale themselves (grade their progress) on various topics and view the progress made. In this report, we describe the design process of our product. It describes various phases of the project including the specification of requirements, the design choices and the testing of our design. The system was designed with the main focus on usability, so that every student and teacher could use the application easily. The Lyceum de Grundel intents for the project to be the pilot phase of an actual product that the school will use in the future.

Contents

1 Introduction

1.1 About the authors

Jasper Bovenkerk is a 3rd year Technical Computer Science student. He has been a TA for 3 modules, so he is familiar with teaching, explaining and grading. Most of his programming experience comes from what was thought during his studies.

Wouter van den Brink is a 6th year Technical Computer Science student. He took the *Leren Lesgeven* (learning how to teach) minor in 2020 and will receive his teaching competence when graduating. Besides learning about didactics and pedagogy, he gained much experience in using LaTeXand academic writing, as the minor is famous for its many reports and assignments. Furthermore, he is very experienced in PHP and the Laravel framework. While he has some experience in frontend development using vanilla Javascript and frameworks like Vue and React, they are not his strong points.

Arda Koç is a 4th year Technical Computer Science student. In addition to his studies, Arda has been a Teaching Assistant for two years. This has led him to work closely with younger students and understand the workload of a teacher. Besides this, Arda has previously worked on projects involving clients outside of the University of Twente. A weak point for Arda is his lack of programming in the past year.

Maarten Meijer is a 3rd year Technical Computer Science student. Next to his studies, Maarten has been a Teaching Assistant for two years. Maarten has a lot of experience in back-end development in languages such as Python, Java, C++ and C#. Maarten also has a lot of experience in Cybersecurity and Deep Learning. Although Maarten has some experience with JavaScript and React, this framework and frontend, in general, are definitely not his strong points. Maarten suspects that especially his general backend development skills and his Cybersecurity skills will come to good use in this project.

David Vos is a 3rd year Technical Computer Science student. He has a lot of experience developing web applications for various personal projects, mainly using JavaScript and Python. Besides this, he has also worked on a few website projects for his study association. He also has experience with creating and deploying Docker containers, which can prove useful when deploying our project on the school's infrastructure.

1.2 Background

Our client H. Hilhorst is the head of teachers of the mathematics faculty at Lyceum De Grundel in Hengelo. Hilhorst is actively working on transforming the educational system of the department according to the findings produced by current research. Popular educational topics include making learning visible as presented by Hattie and Yates in [1]. This practice of *visible learning* has been embedded in various other educational teaching materials, see for example [2], [3].

The scope of this project is that of visible learning through the practice of formative assessment. The teacher gives the student an overview of the learning goals beforehand and continuously evaluates the student's efforts with formative feedback. This means that the work is not graded. Rather, the student receives insightful feedback through which they can improve their work while they are working on it.

Our client would like to utilize the visible learning system by combining learning goals, proofs of learning and formative feedback in a streamlined process. Generally, the iterative system works as follows:

1. The teacher presents an overview of all the learning goals. Every learning goal describes a skill the student must possess and demonstrate in the course.
2. Every learning goal is accompanied by success criteria. These criteria describe the desired end goal. They should help both the assessor and the student to check whether the student has achieved the goal.
3. The teacher gives a lesson and practices a certain skill with the students. The emphasis in this phase is on removing misconceptions and other early mistakes.
4. Students indicate their current competence level by *scaling in*. They indicate their progress in achieving the learning goal on a scale from red (lowest) to green (highest).
5. During the last ten minutes of the class, a final challenge is presented with which students can prove they have achieved the goal.
6. The students submit a picture or scan of their answers. This is a *proof of learning*: it shows they have mastered the learning goal.
7. The students again rate their progress by scaling in. The teacher does the same, using the proof of learning and the experience in class.
8. Students and teacher reflect on the learning process using the feedback cycle described in [4] and [5]. The student actively participates in the learning process, and together it is decided what has to be done.

Note. The process is explicitly different for each student: the above is *not* a recipe one should follow for every student! We apply the principle of differentiated learning: one student will need more practice and assistance, while the other might be able to achieve the learning goal faster or with more independence. As a consequence, our product will have to offer the user many possible ways of using the system, without arbitrary restrictions, e.g. requiring the use of certain features or enforcing deadlines for assignments.

1.3 Previous system

The mathematics department used various rubrics and study guides on paper to implement the described system. The students and teachers would scale in on a printed copy of the study guide. The proofs of learning were submitted online, using Microsoft OneNote.

With everything being on paper, there was a lot of manual labour involved. And students tended to lose or forget their paper study guides. Furthermore, while OneNote was a fairly adequate platform for students to submit their work, there was much unnecessary work involved for the teacher in reviewing and assessing the submitted work.

1.4 Previous Work

Last year, a former project group in the Design Project module developed a system very similar in functionality for the same client. The results were satisfactory for both the developers and the client. However, it was discovered too late that the system was not compatible with the existing IT infrastructure present at the school where the tool was to be used. Consequently, the product was never put into use. It goes without saying that we wish to prevent such a situation, which is why we are in close contact with the IT department of Grundel to verify we will not run into the same issues.

1.5 Source Code

The source code of the frontend and API are available on request. To request a copy of the source code, please contact W. van den Brink by email via wouter@dutch1.nl.

1.6 Glossary

The reader will notice the use of various, sometimes ambiguous terms specific to the problem domain throughout this document. This section briefly introduces, explains, or defines important terms.

Chapter (Hoofdstuk) – A set of learning goals that can be assigned to a group.

Group (Klas) – Consists of a group of students and has chapters assigned to it. Sometimes also referred to as *class*

Learning goal (Leerdoel) – A skill/subject that the student is invited to learn. A learning goal is elaborated by success criteria and optionally other information like exercises.

Success criterium (Succes criterium) – An explanation of things a student should be able to do or know to master a learning goal.

***Proof-of-learning* assignment (Bewijs-van-leren)** – Assignment that challenges the student to upload a picture of a certain exercise to show that the learning goal has been achieved

Scaling in (Inschalen) – Assessing proficiency on a learning goal, by indicating a spot on a bar coloured red to green. Green indicates that the learning goal has been mastered and red means that there is no knowledge about the learning goal.

2 Project Approach

2.1 Introduction

In this chapter, we present the approach with which we will be developing the project. By documenting and agreeing on elements like the workflow and the planning, we can ensure everybody involved, i.e. both client, supervisor, and developers, know what is to be expected at what point. Furthermore, the project approach serves as a tool for everybody to intervene in a timely manner if something were to go wrong, without grave consequences to the product.

We have chosen a project approach inspired by Scrum [6], in which we divide the product into smaller tasks which we will tackle one by one in several sprints.

Wouter van den Brink takes on the role of scrum master. He is tasked with reminding the team of the tasks at hand, reorganizing the backlog when needed and making sure we stay on track with deadlines. We have designated product owners for the various parts of the product. An overview of the parts and their owners are given in Table 2.1.

2.2 Practices

We will divide the project into five sprints of two weeks each, with exception of the first sprint, due to the holiday week. The timing of the sprints and their objectives are presented in Table 2.2. Every sprint, with exception of the first sprint, contains a peer review meeting offered by the module coordinator. Furthermore, we will have a presentation and poster session during the last sprint. These meetings are outlined in Table 2.3. A full overview of our planning is presented in Appendix C.

In the peer review meetings, we are able to share and discuss our current work with fellow students and receive feedback so we can improve our work. Generally, we wish to discuss the output produced in the previous sprint in the peer review meeting of the current sprint.

Part	**Lead**
Frontend	David Vos
Backend	Wouter van den Brink
Design report	Maarten Meijer
Reflection essay	Arda Koc
Poster and presentation	Jasper Bovenkerk

Table 2.1: Various product parts and their owners.

#	Start	End	Objective
1	07-02	25-02	Project Proposal and Design Mock-ups
2	28-02	11-03	Test Plan and Lo-Fi Prototype
3	14-03	25-03	Hi-Fi Prototype and Testing
4	28-03	08-04	Final Product, Presentation Slides and Design Report
5	11-04	22-04	Manual, Poster and Design Report

Table 2.2: The sprints and their main objectives.

#	Date	Topic
1	09-02	Introduction, selection of groups and assignments
2	01-03	Peer review: project proposal and planning
3	15-03	Peer review: requirements and test plan
4	29-03	Peer review: design and first prototype
5	12-04	Presentation
6	20-04	Final poster session

Table 2.3: The various meetings that are part of the Design Project module.

2.3 Reflection Essay

As part of the reflection component of the Design Product module, we will write an essay on ethics in the context of our product. This reflection report attempts to connect our project with one of the ethical topics discussed in the course. The report is to be quite large (approximately 6 pages and 1800 - 2400 words), so we must consider the work required while creating the planning.

We start work on the reflection essay in the third sprint, where we write an outline and divide the tasks among the project members. We aim to finish the reflection essay in the fourth sprint, so we can submit it before the deadline of April 13$^{\text{th}}$ in the fifth sprint.

2.4 Sprints

Every sprint starts by collecting and refining the tasks we wish to execute during the sprint. For example, if we could not finish a task during the last sprint, but we still need to do so, it has to be moved to the current sprint, and we might have to reconsider the prioritization of other tasks in the current and next sprints. The described situation is of course not desired, but it is not unavoidable either.

During a sprint, we will divide the tasks at hand over the developers. Every task can be assigned to one or more team members. A task can generally be worked on anywhere, and due to the current pandemic and personal preferences, it might be favourable to work on one's tasks from home. We will keep in contact with each other through WhatsApp and we will regularly, i.e. at least twice a week, meet each other to work on the project together and to discuss the progress of the project.

Sprint 1: Project Proposal and Design Mock-ups

We will explore the client's desired product and attempt to transform it into a project proposal – this document. The client's desired features are made tangible in the form of user stories and a global description of the product's design. The project proposal is accompanied by rough mock-ups of the final product, so the client can assess whether we have correctly understood the client's intentions.

Sprint Tasks

1. Create user stories
2. Write a global description of the product design

3. Prioritize user stories
4. Describe project approach
5. Create planning

Sprint 2: Test Plan and Lo-Fi Prototype

Together with the client, we will decide on how we will be testing the final product 'in the wild' with the intended users of the product. The test plan is put into practice in the next sprint, using the Hi-Fi prototype. The results of the test plan will help us in refining the final product. Of course, it can happen that the tests highlight key issues in our design or implementation.

Next, during this sprint, we will develop the first lo-fi prototype based on the mock-ups we created in the previous sprint and the feedback we received from our client. The lo-fi prototype will be developed in the target programming language and environment and will be presented to the client for inspection and review.

The lo-fi prototype will focus on two key interactions. First, users of the product should be able to indicate a student's progress of a learning goal by scaling in. This interaction is a major feature of the product and the development should start early so that we can continuously refine the experience.

Similarly, submitting and reviewing proofs of learning is a great interaction for the product, and we must start development early. While the practice of uploading, storing and retrieving files is a solved problem, creating a useful and smooth user interface for both the teacher and the student is not.

Sprint Tasks

6. Create a test plan for ourselves.
7. Create test instructions for participants
8. Implement authentication and authorization
9. Create UI scaffolding[1]
10. Develop scaling interface
11. Implement uploading of proofs of learning
12. Implement retrieving and grading proofs of learning

Sprint 3: Hi-Fi Prototype and Testing

We will now finalize the product by adding other important features, e.g. learning goals, success criteria, overview screens, and class and user management. This will all be developed as a hi-fi prototype.

The prototype is then tested according to the test plan developed in the previous sprint. The test results are documented and we will decide on what improvements and changes we will introduce in the product to create the final product in the next sprint.

Sprint Tasks

13. Implement learning goals
14. Implement success criteria
15. Create student overview screen
16. Create teacher overview screen
17. Create administrator overview screen
18. Implement class management
19. Implement user management

[1] Create building blocks for the various UI elements we wish to use throughout the project.

20. Write reflection essay outline

21. Work on reflection essay

22. Perform user testing

Sprint 4: Final Product, Presentation Slides and Design Report

In this sprint, we finalize the product. Any task which has been prioritized as a Must will have to be finalized in this sprint. The product is polished and any comments and feedback that have been brought up by the client or in the test phase will be taken into consideration.

In the next sprint, we will give a presentation of the product and the project. We will be preparing the presentation and presentation slides, so we can focus on other tasks in the sprint containing the presentation.

Finally, we will begin work on the design report. The design report is a big deliverable that will span this and the next sprint. We will outline the design report and work on the first few chapters. We can use the existing LaTeXtemplate used for our initial project approach report as a basis for the design report.

At the end of this sprint, we introduce a *feature freeze*. The product will not be modified significantly anymore, and no new features can be requested, nor will they be developed. The reader should consider this a friendly warning, as the feature freeze implies a deadline for requested features halfway through this sprint. We might not be able to develop a requested feature right before the feature freeze, depending on the gravity and complexity of the feature.

Sprint Tasks

23. Implement remaining features

24. Evaluate and implement client feedback

25. Evaluate and implement feedback from test results

26. Create presentation outline

27. Create presentation slides

28. Create design report outline

29. Work on design report

30. Work on reflection essay

31. Finish and submit the reflection essay

Sprint 5: Manual, Poster and Design Report

Having finalized the product in the previous sprint, we can now develop a user manual for teachers, administrators and students. The manual will guide the users of the product in using the product, using screenshots and textual instructions. As the client has emphasized the (future) compatibility with the school's IT infrastructure as a very important requirement, we should pay great attention to installation and deployment instructions in the administrator's manual.

Furthermore, our project and product are to be presented in a poster at the final poster meeting on April 20^{th}. The poster should be created and delivered *during this sprint*. The poster is accompanied by a product demo, which we will need to prepare.

Finally, we will finish the design report according to the outline created in the previous sprint. The design report, finished product, manuals and posters are to be submitted to the client, BOZ and the bachelor showcase.

Sprint Tasks

32. Write teacher manual

33. Write administrator manual

34. Write student manual

35. Create poster

36. Prepare demo

37. Finish design report

2.5 Client and Supervisor Meetings

We will hold weekly meetings with our supervisor, M. Huisman, to keep her in the loop of our progress, ask questions and receive feedback when requested. The meetings are supposed to be held at the same time and day of the week, every week. This happens on a best-effort basis.

We wish to hold weekly meetings with our client, H. Hilhorst. We will present our progress, ask questions, receive feedback, and refine the product specification as needed. Furthermore, we will stay in close contact with the school's IT department, to continuously verify that our product will work in the target setting.

2.6 Risk Analysis

Developing a project at this scale does not come without risks. In this section, we present the risks as we predict them in the development of the product, and we describe how we intend to mitigate any damage, delays or other negative consequences if the worst-case scenarios do take place.

2.6.1 Hosting and Deployment Problems

First, we remark that compatibility with the school's IT infrastructure is a very essential requirement for the viability of the product. If we cannot deploy the application for use by the school's teachers and students, the product will never be used and our time spent developing it will be for nothing.

We are in close contact with the school's IT department and wish to start deploying early versions of the system as soon as possible so that we can tackle any compatibility issues we encounter long before the application is to be delivered. Next, most group members are familiar and/or have some experience with the deployment system in use by the school and feel confident we can deploy the project in time. In the worst-case scenario, we could look for alternative hosting options and discuss the costs involved with the school.

2.6.2 Missed Deadlines and Extracurricular Activities

Most group members are active in many committees, are board members of associations or have other responsibilities besides their studies during the project. We need to be mindful of everybody's different schedules and ensure we make the most of our limited available time. We actively mitigate scheduling issues by indicating our weekly availability using LettuceMeet[2]. Using the availability matrix, we can allocate efficient working hours using a shared online calendar. So far this system has proven to work well.

We make extensive use of Trello for task and deadline management. The scrum master regularly checks the team's progress in the shared Trello board and pays attention to upcoming deadlines. Every card has a deadline according to the planning in Appendix C, so that in the worst-case Trello can remind us autonomously.

[2] https://lettucemeet.com/

3 Requirements Specification

3.1 Introduction

This chapter discusses the requirements on which our system has been based. The requirements were initially set out as user stories, which we then prioritized based on the MoSCoW principle. We finalized the requirements at the end of the fourth sprint, where we introduced a feature freeze.

As outlined in chapter 2, we have used the Scrum principle, which divides the project into multiple *sprints*. For each sprint, a set of requirements to work on was chosen. At the end of each sprint, ideally, these requirements would be implemented and finished.

3.2 Description of the Requested System

The system should be a tool that allows both teachers and students to easily document and evaluate the progress of a student. The system should be a website that teachers and students can use on their phones or on their computers. Students can view learning goals and success criteria corresponding to these goals. Students should reflect on their own skills by scaling themselves on a gradient from red to green, indicating to what extent they have mastered the subject. Red means that a student believes they have not mastered the subject, while green means that they have. This scale is explicitly not very objective and clear, so students have some freedom when interpreting the scale.

Teachers should also be able to scale the students so that the student can get important formative feedback. This way, a student can evaluate whether the teacher agrees on how they scaled themselves.

The application should also provide a way for students to prove how much they actually learned. To achieve this, teachers can create assignments where students should submit a picture of their writing.

3.3 Example Workflows

In this section, we will discuss how the system will work using workflows. A workflow is a description of what the user will see and what path a user will follow to accomplish a certain task. We describe workflows for the teachers and the students.

3.3.1 Student

Students will be able to perform three distinct tasks: handing in a proof of learning, looking up information or feedback and scaling in on a learning goal.

The student will always start by signing in to the application using their school-issued email address and password. After successfully authenticating, the student will arrive on a home screen where all their classes can

be viewed. This might be some set of all courses, like math, history and physics (Figure A.2). The student can then click on a class and will go to the specific class, e.g. math (Figure A.3).

On this screen, the student will be able to view the chapters that belong to the class. Teachers can choose to use different classifications to define the boundaries of a chapter. A chapter could for example correspond to a chapter from the coursebook. One chapter will be selected by default. It is possible to switch between chapters using a menu. When a chapter is selected, all learning goals that belong to it will be shown.

Then, for each concept in that chapter, there will be one or more learning goals, e.g. "Linear functions". Upon selecting a learning goal, the success criteria, scaling bar and proof-of-learning assignments belonging to the goal will become visible.

Handing in a proof of learning can be done by clicking on a button. This will give an overview (Figure A.5) of the proof-of-learning assignment with an option to select an image from the device to hand in. In case a proof of learning has already been handed in, it can be viewed, accompanied by any feedback left by the student or teacher.

When selecting a learning goal, the red to green coloured bar for scaling in will be visible, with previous reflection moments of both the teacher and the student visible on the bar. A student can scale themselves in by clicking on the bar at the desired location.

3.3.2 Teacher

We have envisioned five distinct workflows for the teacher. The teacher will be able to create and edit chapters, create classes, change the chapter for a class, view the progress of a class, and review proof-of-learning assignments for a class.

The teacher will always start by signing in using their school-issued email address and password. After successfully authenticating, the teacher will arrive on a home screen (Figure A.6), which displays an overview of the classes taught by the teacher. A button will allow the teacher to create new classes. Students can be added to this class either by uploading an Excel file from Somtoday[1] or by manually adding students using their email addresses and names. Finally, teachers will be able to assign one or more chapters to a class.

There is a sidebar on the left side of the screen, with which the teacher can switch between viewing classes and chapters. The sidebar can also be used to visit the home page. When creating a new chapter, the teacher can immediately start adding learning goals, success criteria and proof-of-learning assignments.

Once a chapter is created, it can be found in the chapter overview. The teacher can then edit or remove the chapter. Note that editing or removing a chapter will not affect the chapters for a class that have already been assigned; those will have to be edited from the class view.

The teacher can use the sidebar to go to the class view again. After clicking on a class there is an option to add chapters or to edit/remove the ones previously assigned (Figure A.11). This editing will only affect the chapter for this specific class.

In addition to being able to create, edit and remove classes and chapters, the teacher also has the ability to view the progress of students from the class view. When a teacher clicks on a class and then selects a specific chapter, it will show the contents of that chapter and an overview of how the students scaled themselves in for each learning goal (Figure A.7). In addition, if there were any proofs of learning to be handed in, the teacher can click on a button to view all the submissions of the students (Figure A.10). In this view, there will be the submitted picture, a bar for scaling in the student and a field to give feedback. The teacher will be able to go through all submissions quickly from this view.

3.4 User Stories

The requirements of the requested system are described below in the form of User Stories. "A user story is an informal, general explanation of a software feature written from the perspective of the end-user. Its purpose is to articulate how a software feature will provide value to the customer." [7] Each story is short and concise so that we can directly use them as sprint tasks. It is easy to see the progress of the development by viewing how many

[1] the school's electronic learning environment

User Stories are met. The User Stories can be divided into stories for teachers, students and administrators. The user stories listed below are suffixed with their prioritization according to the MoSCoW principle.

3.4.1 User management

As any user...

1. I want to be able to log in using my school account from Microsoft. (S)
2. I want to be able to create an account with my school e-mail and a custom password. (M)
3. I want to be able to log in with my created account. (M)

Initially, we will authorize users by using the system database, meaning users will have to create an account with the system. If possible, and if time allows, we will use the school's Microsoft Single Sign On (SSO) system for authentication.

As an administrator...

4. I want to be able to enable and disable accounts for teachers. (M)
5. I want to be able to enable and disable accounts for students. (M)

3.4.2 Class management

As a teacher...

6. I want to have an overview of all my classes, with insightful notifications. (S)
7. I want to be able to create classes. (M)
8. I want to be able to edit and remove classes so that I can fix mistakes if I make them. (M)
9. I want to be able to add or remove students if changes occur during the year. (S)
10. I want to be able to create learning goals for my classes. (M)
11. I want to be able to edit and remove learning goals for my classes, so that I can fix mistakes if I make them. (M)
12. I want to be able to assign success criteria to learning goals for my classes. (M)
13. I want to be able to edit or remove success criteria from learning goals for my classes so that I can fix mistakes if I make them. (M)

3.4.3 Chapter management

As a teacher...

14. I want to be able to create a chapter. (M)
15. I want to be able to edit and remove a chapter so that I can fix mistakes if I make them. (M)
16. I want to be able to import a chapter into my class. (M)
17. I want to be able to create learning goals for a chapter. (M)
18. I want to be able to edit and remove learning goals for a chapter so that I can fix mistakes if I make them. (M)
19. I want to be able to assign success criteria to learning goals in a chapter. (M)
20. I want to be able to edit and remove success criteria from learning goals in a chapter so that I can fix mistakes if I make them. (M)

3.4.4 *Proof-of-learning* assignment management

As a teacher...

21. I want to be able to create a *proof-of-learning* assignment, coupled with a learning goal. (M)

22. I want to be able to edit and remove a *proof-of-learning* assignment, coupled with a learning goal so that I can fix mistakes if I make them. (M)

23. I want to be able to add a solution to a *proof-of-learning* assignment, in the form of a picture, text or PDF file. (C)

24. I want to be able to add a rubric to a *proof-of-learning* assignment, in the form of a picture, text or PDF file. (C)

3.4.5 Reviewing and scaling

As a teacher...

25. I want to see an overview of the progress and status of all learning goals in my class, also showing the date that students scaled themselves. (M)

26. I want to be able to review a specific student in my class for a specific learning goal. (M)

27. I want to be able to review a specific student in my class multiple times for a specific learning goal, keeping track of the date I scaled them. (S)

28. I want to see an overview of the progress and status of all *proof-of-learning* assignments in my class. (S)

29. I want to be able to review a specific student in my class for their proof of learning. (S)

30. I want to be able to review a specific student in my class multiple times for their proof of learning, keeping track of the date I scaled them. (C)

31. I want to be able to draw feedback on the uploaded proof of learning of a student. (C)

32. I want to be able to see a rubric when reviewing any student in my class for a learning goal or a *proof-of-learning* assignment. (C)

33. I want to be able to undo my review of a learning goal or a *proof-of-learning* assignment if I decide so within a minute. (C)

34. I want to be able to see the comment (feedback) from a student on their learning goals and *proof-of-learning* assignments. (C)

35. I want to be able to comment (give feedback) on a learning goal or *proof-of-learning* assignment for a specific student. (S)

As a student...

36. I want to be able to have an overview of the *proof-of-learning* assignments I should still hand in. (C)

37. I want to be able to see the current status of my progress in a specific class, seeing how I scaled myself on different learning goals and *proof-of-learning* assignments. (C)

38. I want to be able to see my reflections and feedback from previous years. (C)

39. I want to be able to reflect on different learning goals by scaling myself on a gradient from red to green. (M)

40. I want to be able to reflect on different learning goals multiple times, keeping track of the dates that I scaled myself. (S)

41. I want to be able to upload my proof of learning as a picture, text or PDF file. (M)

42. I want to be able to upload a proof of learning multiple times on the same assignment. (C)

43. I want to be able to reflect on my proof of learning by scaling myself on a gradient from red to green. (C)

44. I want to be able to reflect on my proof of learning multiple times, keeping track of the dates that I scaled myself. (C)

45. I want to see how my teacher scaled me on a learning goal or a *proof-of-learning* assignment, showing which teacher scaled me and when. (M)

46. I want to be able to see a rubric when reflecting on myself for a learning goal or a *proof-of-learning* assignment. (C)

47. I want to be able to undo my reflection of a learning goal or a *proof-of-learning* assignment if I decide so within a minute. (C)

48. I want to be able to see the comment (feedback) from a teacher on my learning goals and *proof-of-learning* assignments. (S)

49. I want to be able to comment (give feedback) on my learning goals and *proof-of-learning* assignments. (C)

3.5 Prioritization

We use the MoSCoW principle to prioritize the various user stories. We classify user stories as *must*, *should*, *could* and *will not*. User stories classified as *must* are the parts that are required for the application to fulfil its purpose. *Should*-stories are still very important parts of the applications, but the application can function without them. Stories in the *could* class are features that are mostly optional and will only be implemented if we have the time to spare. Finally, those stories in the *will not* class are features that are outside the scope of the project, either due to time or complexity constraints.

Must

1. As any user I want to be able to create an account with my school e-mail and a custom password.

2. As any user I want to be able to log in with my created account.

3. As an administrator I want to be able to enable and disable accounts for teachers.

4. As an administrator I want to be able to enable and disable accounts for students.

5. As a teacher I want to be able to create classes.

6. As a teacher I want to be able to edit and remove classes if changes occur during the year.

7. As a teacher I want to be able to create learning goals for my classes.

8. As a teacher I want to be able to edit and remove learning goals for my classes so that I can fix mistakes if I make them.

9. As a teacher I want to be able to assign success criteria to learning goals for my classes.

10. As a teacher I want to be able to edit or remove success criteria from learning goals for my classes so that I can fix mistakes if I make them.

11. As a teacher I want to be able to create a chapter.

12. As a teacher I want to be able to edit and remove a chapter so that I can fix mistakes if I make them.

13. As a teacher I want to be able to import a chapter into my class.

14. As a teacher I want to be able to create learning goals for a chapter.

15. As a teacher I want to be able to edit and remove learning goals for a chapter so that I can fix mistakes if I make them.

16. As a teacher I want to be able to assign success criteria to learning goals in a chapter.

17. As a teacher I want to be able to edit and remove success criteria from learning goals in a chapter so that I can fix mistakes if I make them.

18. As a teacher I want to be able to create a proof-of-learning assignment, coupled with a learning goal.

19. As a teacher I want to be able to edit and remove a *proof-of-learning* assignment, coupled with a learning goal so that I can fix mistakes if I make them.

20. As a teacher I want to see an overview of the progress and status of all learning goals in my class, also showing the date that students scaled themselves.

21. As a teacher I want to be able to review a specific student in my class for a specific learning goal.

22. As a student I want to be able to reflect on different learning goals by scaling myself on a gradient from red to green.

23. As a student I want to be able to upload my proof of learning as a picture, text or PDF file.

24. As a student I want to see how my teacher scaled me on a learning goal or a *proof-of-learning* assignment, showing which teacher scaled me and when.

Should

25. As any user I want to be able to log in using my school account from Microsoft.

26. As a teacher I want to have an overview of all my classes, with insightful notifications.

27. As a teacher I want to be able to add or remove students if changes occur during the year.

28. As a teacher I want to be able to review a specific student in my class multiple times for a specific learning goal, keeping track of the date I scaled them.

29. As a teacher I want to see an overview of the progress and status of all *proof-of-learning* assignments in my class.

30. As a teacher I want to be able to review a specific student in my class for their proof of learning.

31. As a teacher I want to be able to comment (give feedback) on a learning goal or *proof-of-learning* assignment for a specific student.

32. As a student I want to be able to reflect on different learning goals multiple times, keeping track of the dates that I scaled myself.

33. As a student I want to be able to see the comment (feedback) from a teacher on my learning goals and *proof-of-learning* assignments.

Could

34. As a teacher I want to be able to add a solution to a *proof-of-learning* assignment, in the form of a picture, text or PDF file.

35. As a teacher I want to be able to add a rubric to a *proof-of-learning* assignment, in the form of a picture, text or PDF file.

36. As a teacher I want to be able to review a specific student in my class multiple times for their proof of learning, keeping track of the date I scaled them.

37. As a teacher I want to be able to draw feedback on the uploaded proof of learning of a student.

38. As a teacher I want to be able to see a rubric when reviewing any student in my class for a learning goal or a *proof-of-learning* assignment.

39. As a teacher I want to be able to undo my review of a learning goal or a *proof-of-learning* assignment if I decide so within a minute.

40. As a teacher I want to be able to see the comment (feedback) from a student on their learning goals and *proof-of-learning* assignments.

41. As a student I want to be able to have an overview of the *proof-of-learning* assignments I should still hand in.
42. As a student I want to be able to see the current status of my progress in a specific class, seeing how I scaled myself on different learning goals and *proof-of-learning* assignments.
43. As a student I want to be able to see my reflections and feedback from previous years.
44. As a student I want to be able to upload a proof of learning multiple times on the same assignment.
45. As a student I want to be able to reflect on my proof of learning by scaling myself on a gradient from red to green.
46. As a student I want to be able to reflect on my proof of learning multiple times, keeping track of the dates that I scaled myself.
47. As a student I want to be able to see a rubric when reflecting on myself for a learning goal or a *proof-of-learning* assignment.
48. As a student I want to be able to undo my reflection of a learning goal or a *proof-of-learning* assignment if I decide so within a minute.
49. As a student I want to be able to comment (give feedback) on my learning goals and *proof-of-learning* assignments.

Will not

We have not classified any of the requirements as a *will not* requirement – we wish to (attempt to) implement every user story.

3.6 Performance Requirements

In this section, we discuss some requirements that explain how the system should perform. The performance requirements will be tested with user testing and automated testing.

There are a few important requirements for this system, which can be found in our performance requirements. It should be easy to learn, as a steep learning curve would deter people from starting to use this tool. It should also be quick to use, as it was created with the intention to replace a paper system that was too inefficient and burdensome for many teachers to adopt. A previous project has had difficulties due to incompatibility with the school's IT system, we should learn from these previous mistakes.

The considerations above resulted in the following requirements:

1. 80% of the students should be able to intuitively use the application on the desktop browsers Google Chrome, Safari, Edge and Firefox without further explanation.
2. 80% of the students should be able to intuitively use the application on the mobile browsers Google Chrome and Safari without further explanation.
3. 50% of the teachers should be able to intuitively use the application on the desktop browsers Google Chrome, Safari, Edge and Firefox without further explanation.
4. A student should be able to evaluate a learning goal within 5 seconds.
5. Uploading a *proof-of-learning* assignment should take no longer than 2 seconds.
6. The application should be compatible and function as intended on the IT infrastructure of Lyceum de Grundel.
7. The system should be defended against the top ten most common web application security vulnerabilities, as determined by OWASP[2].

[2] https://owasp.org/www-project-top-ten/

4 Global Design

4.1 Introduction

In this section, we will discuss our design. We will first give an overview of the entire application and then we will explain the choices we made regarding specific aspects

4.2 Global Design Choices

The goal of our design is to create an application that can be easily used on virtually any digital device that a student or teacher owns, i.e. desktop computers, laptops and smartphones. Besides this, the system should be intuitive in its use and additionally, we aim to keep the system to be as simple as possible, avoiding creating something complex but rather developing an application that focuses on the task of tracking learning progression without many extra features that would turn it into a larger student management application.

In order to achieve these goals, we made some preliminary choices, mainly related to our choices of languages and frameworks, as outlined in section 4.3. The global design of the system is outlined in section 4.4.

4.3 Preliminary Design Choices

Because we needed the product to be actually compatible with our client's IT infrastructure, we have sought contact with the school's IT department and inquired about the technical requirements and limitations of the school's infrastructure.

The school makes use of a Kubernetes environment in which Docker containers can easily be deployed and run. Because of this, there are not many limitations on our choice of language or framework for the backend, as long as we can deploy it in a container.

Therefore, we settled on languages and frameworks that our group members were already familiar with and that we deemed suitable for a project such as this.

4.3.1 Web application

In order to implement our project, we chose to develop a web application. We did this mainly because students and teachers need an easy way to access our application. With a website, there is no need for them to install any additional software and they are able to access the application from any device. Of course, this means that our application will also be compatible with mobile users since students could also use their phones to access the application.

4.3.2 Backend

Laravel

Initially, we considered using Python and the Django web application framework. However, while most group members have some experience with this stack, we do not consider ourselves skilled enough to create complex web applications such as the product described in this report using Django.

On the other hand, Wouter is very experienced in PHP and the Laravel framework. The other group members believe they are experienced enough to be able to learn the basics of the language and framework quick enough to be able to provide a meaningful contribution to the backend.

Thus, by using Laravel, we mitigate the risk of not being able to complete the project due to an overall lack of knowledge and experience, at the 'cost' of making Wouter mainly responsible for the development of the backend.

File storage

The IT department has asked us to use a solution like Google or Amazon S3 Buckets in order to store user uploaded files since that is easier in their environment compared to using disk storage. We settled on using Amazon S3 since Wouter has more experience with that.

4.3.3 Frontend

JavaScript

As this is a web application, JavaScript is an obvious choice as this is the main language used for developing websites. We also chose JavaScript so that we could use the React framework. Because JavaScript code executes on the side of the client this could be exploited for malicious purposes. This means that we have to be extra careful with our security implementation to ensure a safe user experience for both teachers and students.

React

React is a popular framework to create frontends. The group members working on the frontend were already familiar with it, hence our choice of this framework.

4.4 System overview

4.4.1 General

Login

The initial page that any user sees after loading the website is the login page. Users can log in easily with an e-mail and password or can create a new account, this account has to be verified using their school e-mail. For this, a verification link will be sent to the e-mail the user has provided.

4.4.2 Students

Home

The first page that a student sees after logging in is a homepage that lists proof-of-learning assignments with upcoming deadlines in addition to all the courses that they are enrolled in. They can navigate to a specific assignment or to one of their courses. The latter can be done by clicking on a course on the homepage or by using the sidebar menu.

Courses and chapters

Once a student has navigated to a course or "vak", they have the option to select the chapter ("hoofdstuk") that they want to look at and they will be displayed the learning goals ("leerdoelen") of that chapter.

Learning goals

Each learning goal can have a time period, description and the option for both student and teacher to "scale in" (grade) and furthermore consists of a set of success criteria and/or proof-of-learning assignments. After expanding a learning goal by clicking on it, these criteria and assignments are shown.

Besides being able to navigate to the assignments in order to hand them in or view their feedback, students can also grade themselves on each learning goal. It is possible to do this multiple times, by simply clicking on the coloured bar that corresponds to their skill regarding a goal. The bar will show the grades that both their teacher has given them as well as the grades that they have given themselves.

Assignments

After navigating to an assignment, a more detailed description of the assignment is shown, as well as a button leading to a dialogue where the student can hand in the assignment. The student has the option of handing in either some images or some text. After handing in, the student sees their handed in solution and a grading bar as well as a comment section where they and their teacher can comment on the submitted work. An assignment can be handed in multiple times.

4.4.3 Teachers

Home

After logging in, the teacher gets a list of the classes that they teach. They can navigate to these classes by clicking on them or they can edit, delete or add classes. They can also use the sidebar menu to navigate to classes or to chapters. More information about this can be found in subsection 4.4.3 and subsection 4.4.3.

Viewing classes and chapters

Once navigated to a class, teachers get the same view as students get - as outlined in subsection 4.4.2. There are a few differences, however. Firstly, a teacher first gets an overview of the whole class, i.e. the grading bars show little dots representing the grades of each student. The teacher can navigate to an individual student by searching for their name or selecting their name in a list. After this, the grading bars show the student's grades for each learning goal, similar to a student's own view.

A teacher also has buttons to add or remove assignments and a button to edit the learning goals of a chapter, as outlined in subsection 4.4.3.

Assignments

A teacher can view an assignment by navigating to it from the chapter page. They will start with a view of the submissions from all students, through which they can scroll. They can also go to the submissions of a specific student by searching for their name or selecting it from the list. A teacher is also able to add their own solutions to an assignment, which can help with grading or can allow the student to grade themselves.

Teachers also have the option to correct submissions by drawing on them. They can additionally use the comment section in order to give feedback to a student.

Editing chapters

In the application, two kinds of chapters exist. One that is shared between teachers and is not linked to a class and one that is not shared with other teachers and is linked to a specific class. Teachers can edit the shared chapters by going to the page "Hoofdstukken" from the navigation bar. In there, teachers can create and edit these shared chapters. Teachers can edit chapters that are not shared by using a button at the bottom of the chapter page.

For chapters, teachers have the possibility to create learning goals and add success criteria to these learning goals. Teachers can change the order of these by dragging them through the list. There is also the possibility to change the description of both success criteria and learning goals. Learning goals can also have two dates, in between which the learning goal will be active and should be "learned" by the students. Teachers are also able to add a rubric to a learning goal, or even to the whole chapter. These rubrics can be uploaded as image files. The teacher is free to use these rubrics in the way they like. For example, they could be used to show a

schedule, show requirements or show how the scaling bar should be interpreted. Furthermore, teachers can edit with whom they would like to share the chapter.

Managing classes

Teachers have the ability to add, change and remove their classes. We have chosen to make the teachers responsible for as many parts of the application as possible so that our client does not need specific administrators for the system. Teachers are able to add shared chapters to a class, at which moment that chapter will be copied and attached to that class. Teachers are given the right to add and remove students from their classes, which means that teachers also have the responsibility to set up their classes if they want to use the application. To make this process easier, teachers can use an Excel file in a specific format - which the school already uses - to import all students into a class.

4.4.4 Admins

Admins have the same rights as teachers, but can additionally manage the accounts of students and teachers.

Managing users

An admin has an extra page called "Beheer" (manage), where they can view and search through a list of both students and teachers. Here they can delete users if needed.

5 Detailed Design

5.1 System Description

The system is implemented as a web application with a React frontend and a Laravel backend. The backend is a REST API and we use OpenAPI to document the available endpoints. The frontend communicates with this API and correctly displays and sends information from and to the backend API.

5.1.1 Data Model

The application's data is organized according to the class diagram in Figure 5.1.

In this class diagram, every user of the application is a `User`. Next, notice how every administrator is a teacher. This is on purpose, as our client has indicated this is generally how the system will be used at the school. If the system is to be used by an administrator who is not a teacher, they will simply be assigned no classes.

The inheritance constraints imposed by the various relations, like only teachers being able to author chapters, is not enforced on the database level, but rather in the application logic using policies. These policies are further explained in subsection 5.1.3.

A class is called a `Group` to prevent issues with the reserved keyword `class` in the source code of the application. We consistently use this naming scheme everywhere, including the API documentation, to prevent developer confusion.

A `Chapter` represents a chapter within a course. We did not add separate logic for assigning chapters to subjects, as most teachers generally teach one or maybe two subjects. Of course, this functionality can be added later on if it is deemed desirable. Every chapter has zero or more learning goals and every learning goal has zero or more success criteria. On these models, we keep as much fields as possible nullable, so the user can use as many or as little features as they wish when using the system. For the same reason, we do not enforce associating success criteria with learning goals.

The practice of commenting on and scaling entities in the application is implemented by two abstractions. First, every entity that is supposed to be commented on or scaled is a subclass of `Reviewable`. Second, scales and comments are subclasses of `Review`. This future-proofs the system, as we can now easily introduce more entities that can be commented on or scaled, and we can introduce more entities representing a review should the need arise.

We keep track of uploaded files using the `File` class. Should a user choose to simply enter some text in a form field, the text is directly stored in the database to spare a round trip to the external storage provider. Otherwise, the file is sent to the storage provider and metadata is stored in the database, like the file path and its MIME type. An enumeration dictates whether the file is an uploaded file or a string in the database.

Retrieving uploaded files is made possible using temporary, signed URLs. The backend either generates a signed URL (in the case of local storage drivers) or requests from from the storage provider and presents this URL in the API. Retrieving files directly by their primary key is not possible, rather, the file is included in the response

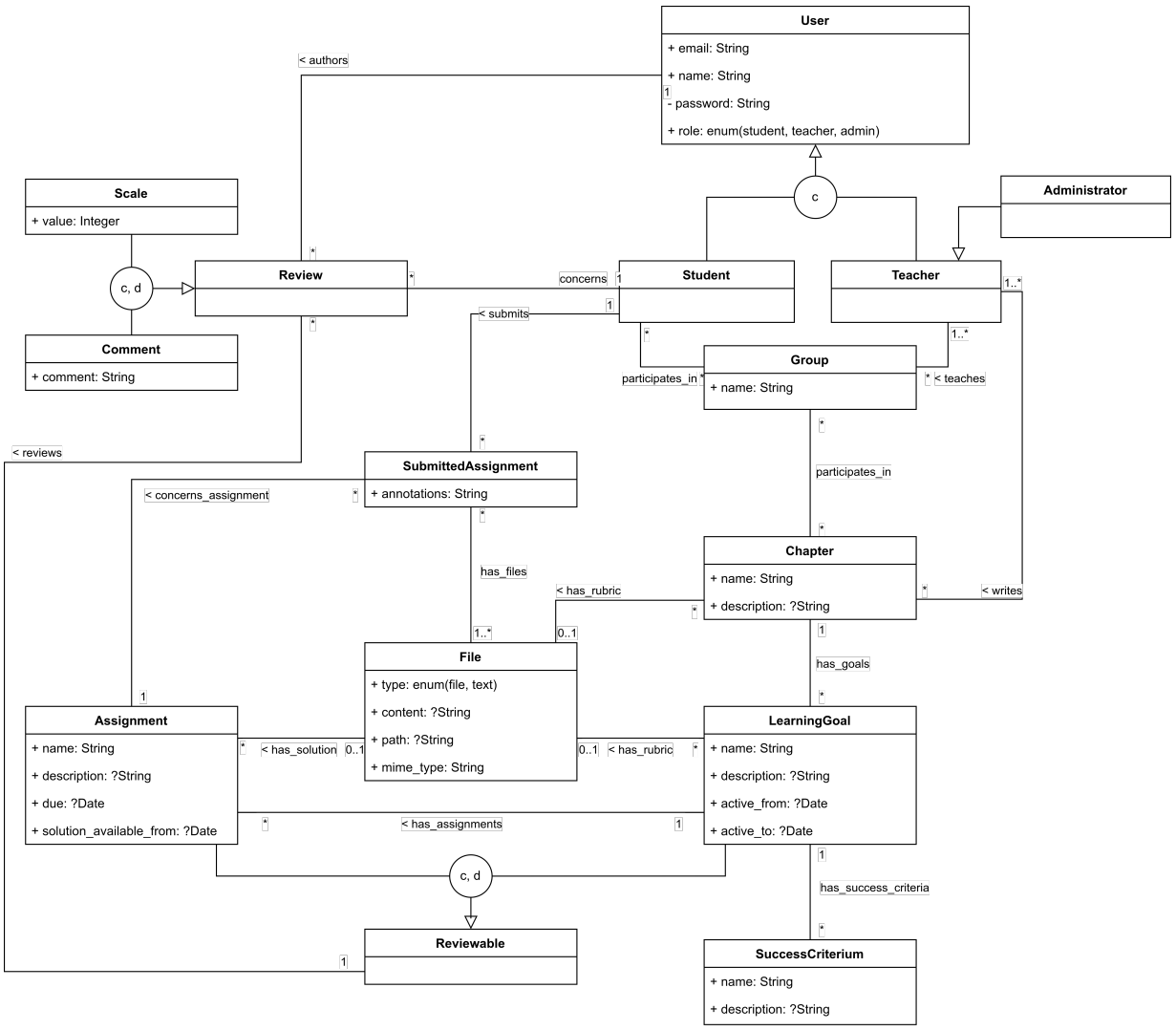


Figure 5.1: Class diagram for the database

payload of other API calls. For example, when retrieving an assignment, the file with its URL and metadata is returned when the solution is present and available.

5.1.2 API Specification

We use OpenAPI[1] to specify the API and generate documentation for the frontend developers. We intended on using code generators to develop the client library for the API, but this proved to be quite difficult as both the OpenAPI library used by the backend and the various code generation libraries we tried contained various bugs.

Eventually we settled on creating our own client library by developing some generic request and response scaffolding and manually adding the various endpoints based on the documentation.

Writing the OpenAPI specification was a tedious job, but by using a library[2] we could use Doctrine annotations and PHP attributes to generate the specification. Despite not being able to use the code generation usually offered by OpenAPI, writing the documentation in this manner proved to be very useful as it made it very clear to the developers what endpoints were available and how they worked.

5.1.3 Security

Authentication

We use the default authentication driver provided by Laravel. We use Laravel Sanctum[3] to add stateful authentication to the API. This package is specifically designed for single-page applications like ours and offers CSRF protection out of the box.

Authorization

Laravel policies are used to determine when a user will be able to perform certain actions. Policies are essentially methods receiving the authenticated user and one or more models on which an action is to be performed. The method returns a response object either approving the action or denying it, along with a reason why the action cannot be performed by this user or on these models. The policies are centrally managed within one namespace according to the SOLID principles. One drawback of policies is that the developer must explicitly use them whenever an action should be authorized. Feature tests are used to verify authorization behavior and to find regression bugs.

Injection

As the frontend makes use of the React framework, sanitisation and protection against JavaScript code injection automatically take place in any situation where user-submitted text is displayed on the webpage. The API uses JSON to encode responses. The encoding takes place using Laravel's own library methods. Thus, it is not possible to inject code or data unsafely in the backend.

Vulnerability of Components

In order to ensure that the chance of vulnerable or outdated components used in the project stays low, the project uses recent and stable versions of all libraries. Were the project to have continued development, as outlined in section 9.2, the developers need to ensure that components stay updated and known vulnerabilities in libraries that are used are circumvented.

5.2 Design Choices

There were a few large choices that were made during the design of the system.

[1] https://swagger.io/specification/
[2] https://zircote.github.io/swagger-php/
[3] https://laravel.com/docs/9.x/sanctum

5.2.1 Scope limitations

Early on in the project, it was decided that we would limit the scope of the application purely to the tracking and displaying of learning progress and the accompanying proofs of learning. This was decided in order to keep the application simple and straightforward. The goal of the project is not to create an alternative to existing student databases like Somtoday or Magister.

Since the application focuses on one goal, that goal can be accomplished solidly whereas the danger of having a large application that does everything poorly arises if the decision to add many more features is made. This design choice is the reason that features such as graphs or other forms of statistics of students are not a feature of the product.

Meanwhile, it was decided to keep the application as limitless as possible for the teachers that will use it. A lot of features have the option to add, for example, a deadline or description, but these things are not required. There are also no limits on, for example, what the description or rubric of a learning goal should exactly be, rather these things are left to the preference of the teacher. This way, they are free to use the application in a way that fits their teaching style and the learning style of their class without having to fight with the project and hopefully limiting the number of future features that could be requested.

5.2.2 Focus on scaling bar

Another design decision that was made has mainly to do with the visual design of the application. Namely, the focus on the coloured bar that is used for scaling in students. It was clear from the conversations with the client that they liked to focus on this scaling bar since it is the main point of the concept that the client wanted and the main point of the product. Therefore, it was decided to keep the colours and design of the other parts of the application simple and clear. That way, the scaling bar stands out to the user.

5.2.3 Roles of admin and teacher

From conversations with the client it became clear that, while an admin role - that can add or remove users if needed - would be helpful, it would very much be preferred if a teacher is able to do almost everything, for example adding other teachers or students to a class, themselves. This way, teachers can easily use the system without having to wait for management or an administrator to add users or classes.

6 Testing

6.1 Test Plan

For our testing, we relied on unit tests, feature tests, and usability tests. The combination of these tests gave us a good overview of the functionality, practicality and correctness of our product.

6.1.1 Unit Tests

Aside from the data models, the API does not contain much classes with specific functionality we should test with unit tests. We created unit tests to verify that the model factories work as intended. This is especially important for regression testing, as the data model has been changed quite a lot while developing the product, and it is easy to forget to update the model factories when changing the models.

Furthermore, we created unit tests for the specific methods added to the models that are not implemented by Eloquent ORM. For example, the `User` model has a method `studies(Chapter $chapter): bool` to query whether the user studies a certain chapter, i.e. they are in a group which studies the chapter. This method was implemented without calling parent Eloquent methods, and thus its behavior is tested with a unit test.

6.1.2 Feature Tests

We created feature tests for the various API endpoints. The feature tests use an in-memory database and model factories to generate mock data and use Laravel's HTTP testing API to query the API. The tests not only verify correct behavior and the presence or absence of side effects, but also provide regression testing. One downside of this method of testing is that the testing environment uses a different database engine (SQLite) than the production environment (MySQL). These databases have some minor differences in what they do and do not allow in terms of constraints. Thus, it might be the case that a feature test passes in the test environment, but would fail in the production environment.

6.1.3 User testing

We performed two iterations of user testing. For each iteration, we had a version of our hi-fi prototype ready so that we could test whether the user interface and system worked as intended for the target audience. However, it was also an important moment to pinpoint possible improvements to the interface or other aspects of the system.

We collected data in two ways. First, we let the intended users of the system navigate and use the product according to a predetermined list of tasks. The users used fake, but plausible data. This way, we prevented issues due to privacy concerns, while keeping the experience realistic. The users were asked several closed questions while using the system.

Second, at the end of the usability test, we asked users several open questions about how they experienced using the system. They were given the opportunity to describe what they liked about the system, but also, more importantly, what they did not like about the system.

The first iteration took place on the 24th of March. Here we had 2 teachers and approximately 25 students answer questions about our hi-fi prototype. The 2 teachers were our client, Hans Hilhorst, and a teacher, Tjark Blokzijl, at Twickelcollege Hengelo, who was also involved in the development. The students were from a class of Tjark Blokzijl. The instructions and questions for this user testing can be found in section B.1.

The second iteration took place on the 5th of April. This time we chose not to include students and focused on the teachers instead. Our main reasons for this were that the teachers have significantly more interactions with the system and that we mainly made changes to the part of the system used by teachers since last time. For this user test, we tested with the same teachers as the previous time, as well as an additional teacher that had never seen the system before. The instructions and questions for this user testing can be found in section B.2.

We did not do any user testing for the admin role, since this role was going to be very small.

6.2 Results

The user testing proved to be very useful, we did find quite a lot of improvements that could be made.

The results of the first user testing were quite positive. In general, both the students and teachers found most of the interface quite clear, however, there were a few parts where the system was not as self-explanatory as we had hoped. To combat this we added some small explanation texts behind a question mark button. It also turned out there were a few more bugs in the system that we had not yet come across. The teachers also pointed out a few more features that could be added.

The results of the second user test were also quite good, however since we had extended the app a bit more this time and we had a participant who had never seen the system before, there were some more things unclear. We wanted to address some of those in the application, but some of the things that were unclear applied to the concept as a whole, so we figured they would be more useful in the teacher manual.

7 Discussion

7.1 Introduction

In this chapter, we will evaluate on the way our Design Project went during the last ten weeks. More specifically, we will look back on how our initial planning panned out, how the eventual division of tasks was, and how we worked together as a team during this project.

7.2 Project approach

The Project Approach was written at the start of the project, so some things in planning turned out best to be handled differently. This section can be seen as an evaluation of the approach we had at the start, what changed and what could have been better.

7.2.1 Practices

We have kept quite well to the practices we decided beforehand. We have used Scrum [6] the whole project, and we all took responsibility for the product parts that we made ourselves owners of.

7.2.2 Sprints

We tried to stay with the Sprint Tasks that we had written at the start of the project, and for the most part, we could do this. The only thing that differed from our initial approach was that we have kept on implementing the remaining features up to and including the very last sprint when we had planned to stop implementing anything after sprint 4.

7.2.3 Risk analysis

The main risks that were identified were difficulties with deployment and hosting and time problems due to extracurricular activities of the group members. We were right in identifying the risks for the project since these indeed turned out to be the two biggest risks. We have dealt very well with the hosting and deployment problems since we knew to look into them from the start.

Furthermore, we have planned well to make sure we all could combine the project with extracurricular activities. We did not use LettuceMeet, but we just planned work sessions by mouth and over WhatsApp. We used Trello a lot to keep track of all the deadlines, and we did not miss any.

7.3 Planning

Our planning was quite strict. We had divided the weeks into sprints, with one sprint consisting of two weeks. Every sprint was filled with objectives that we had to complete to get further in completing our product. Initially, we were able to follow the planning very well. However, during sprint two and the beginning of sprint three, we faced several setbacks. Firstly, due to a COVID-19 outbreak amongst every member of the project team, we started to fall behind during sprint two. This was unfortunate as our planning was quite strict, but nothing can be done about illness. However, we made our working days a bit longer during sprints two and three to make up for the lost time due to illness. Secondly, we realized at the end of the first sprint that we needed more project members working on the front-end than we had initially thought. This meant that fewer project members were working on the back-end. This resurfaced during the third sprint when we faced some issues with the back-end. Only having one person primarily on this made it very difficult to deal with setbacks as a project group.

In the future, it would be much better to assume things will go wrong during the project and anticipate this when making the initial planning. That way there is some leeway when things do indeed go wrong. In addition, it would be more efficient to have multiple project members working on different parts of the project. During our project, we assigned each member a task that they would supervise. However, some tasks only had one member who primarily worked on them. This was fine for us, since we all agreed to it, but for future work, it would be much better if each task had two project members responsible for it. This way, when one of the project members, who is responsible for that part, has fallen ill or faces a setback of any kind the other project member can take over or help where needed. Such a way of working would be an improvement in our opinion.

7.4 Responsibilities

As shown in Table 2.1, every project team member had a task that they were the lead on. The division was made based on each project member's preference. In the end, this division helped as we had one person who would always have an overview on his specific responsibility, even if other project members were also working on it. For future projects, we recommend keeping this structure as it works well for projects which have a lot of different facets and deliverables to it.

7.5 Team Evaluation

From the very first day, the communication between each other went very well. We created a Whatsapp group for internal communication and a Whatsapp group for communication with our client, in addition to using email. We had meaningful discussions with respect to the overall design of our product and how it should look. We tried to divide all of the tasks equally so that everyone had enough work during the development of the application. Furthermore, if project members were late or could not make it on a specific day, we always made sure to communicate this clearly. If such events occurred we would always make sure to help each other out as each of us had a lot of things going on besides our studies. In addition, we created a Trello board where we would put down all the tasks that had to be done during a sprint. It also allowed us to assign project members and deadlines to specific tasks, which helped in keeping everyone up to date on how much progress had been made.

Before we started the project we all wrote down our strong points and our weak points. This really helped in dividing the tasks, because we tried to play to everyone's strengths. In the future, repeating this process is a good idea. Furthermore, all of us being together in a single room physically most of the time helped a lot with the teamwork. In our opinion, this simply worked better than online sessions.

In addition, the communication with the client went very well. We planned weekly meetings with him during which we kept him updated on our progress. Also, during these meetings, we would show him the latest version of our product, which we would give feedback on. This continuous cycle of feedback helped us greatly in changing the product to our client's needs. The communication with the supervisor also went well. We tried to plan in weekly meetings and this worked most of the time, with exceptions such as holidays aside. During these meetings, we would show the latest version of our application and give a short summary of our progress regarding the project, the dynamics within the team, and the communication with our client. All of this went very well.

7.6 Hosting

Initially, we decided to eventually host the project on the servers of Doop, a web hosting company that hosts various services used by Carmel. Their servers run a Kubernetes platform with Docker images. However, when we started creating a staging environment on Wouter's servers, we realized, and confirmed with the client, that it makes more sense to host the production website on Wouter's servers as well. This way, we can easily deploy new versions of the application when needed and fix issues more quickly. During the project, we talked about this with the client extensively and they agreed to continue using Wouter's infrastructure to host the website.

7.7 Conclusion

In conclusion, many facets of our design project went well. The internal communication, work ethic, and communication towards both client and supervisor went smoothly. However, for future projects, it would be much better if the planning was less strict and allowed for some leeway. This is because a lot can happen during a sprint. Deadlines might not be met, unforeseen bugs might appear during development or project members can be sick. Keeping all of this in mind, it is better to go for lighter sprints during the development of the project. However, overall our teamwork and work on the product went fine.

8 Conclusion

8.1 Introduction

In this chapter, the results of this project and the completion of requirements will be discussed.

8.2 Conclusion

We think that we have delivered a product that matches our client's wishes. The project is ready to start a more extensive testing phase where our client will start using the application with some of his classes. During this period it will quickly become apparent if our application matches the expectations and can actually be used on a larger scale.

The aim is that, if the large scale tests are promising, the application will be used by more and more teachers at the Grundel (or other schools in their organisation). During this process, most likely bugs will be found and ideas for improvement will arise. We hope that we can come to an arrangement with Grundel to continue working on the application.

8.3 MoSCoW Prioritization

As visible in the list below most of the requirements that were set were completed. Some of them were not completed because they turned out to be no longer applicable, others would take too much time to implement within the scope of this project. We only present the requirements that were not met or have some remark in the list below. Of the 25 requirements marked as *must*, only 4 have some remark, the others were completed. For the 9 requirements marked as *should*, there were remarks at 5. There were 16 requirements marked as *could*, of those 8 had some remarks on them.

Must

- ✘ **As an administrator I want to be able to enable and disable accounts for teachers.**

  To keep the role of the administrator as limited as possible, the enabling and creating of accounts is done by the users themselves. The accounts are then validated by sending a confirmation to the e-mail.

- ✘ **As an administrator I want to be able to enable and disable accounts for students.**

  To keep the role of the administrator as limited as possible, the enabling and creating of accounts is done by the users themselves. The accounts are then validated by sending a confirmation to the e-mail.

- ✔ **As a student I want to be able to upload my proof of learning as a picture, text or PDF file.**

  We chose to only allow pictures and text since that proved easier for some parts of our implementation.

Should

✘ **As any user I want to be able to log in using my school account from Microsoft.**

We chose to stick with just using the school e-mail for logging in. It was not possible within the timeframe of the project to switch to Mircosoft Single Sign-On.

✘ **As a teacher I want to have an overview of all my classes, with insightful notifications.**

The teacher does have an overview of all their classes, however, we did not implement any notifications.

✘ **As a teacher I want to see an overview of the progress and status of all *proof-of-learning* assignments in my class.**

The teacher will be able to find all the information concerning a proof-of-learning at the proof of learning itself, but there is no overview for all proofs-of-learning and no status available.

✔ **As a teacher I want to be able to comment (give feedback) on a learning goal or *proof-of-learning* assignment for a specific student.**

We did not add comments on learning goals.

✔ **As a student I want to be able to see the comment (feedback) from a teacher on my learning goals and *proof-of-learning* assignments.**

We did not add comments on learning goals.

Could

✔ **As a teacher I want to be able to add a solution to a *proof-of-learning* assignment, in the form of a picture, text or PDF file.**

We chose to only allow pictures since that proved easier for some parts of our implementation.

✔ **As a teacher I want to be able to add a rubric to a *proof-of-learning* assignment, in the form of a picture, text or PDF file.**

We chose to only allow pictures since that proved easier for some parts of our implementation.

✘ **As a teacher I want to be able to undo my review of a learning goal or a *proof-of-learning* assignment if I decide so within a minute.**

We did not have time to implement any form of undo.

✔ **As a teacher I want to be able to see the comment (feedback) from a student on their learning goals and *proof-of-learning* assignments.**

We did not add comments on learning goals.

✘ **As a student I want to be able to see the current status of my progress in a specific class, seeing how I scaled myself on different learning goals and *proof-of-learning* assignments.**

This information can be found at the specific learning goals and proof-of-learning assignments, but we did not make an overview for this.

✔ **As a student I want to be able to see my reflections and feedback from previous years.**

This will still be visible as long as teachers do not delete classes from older years.

✘ **As a student I want to be able to undo my reflection of a learning goal or a *proof-of-learning* assignment if I decide so within a minute.**

We did not implement any form of undo.

✔ **As a student I want to be able to comment (give feedback) on my learning goals and *proof-of-learning* assignments.**

We did not add comments on learning goals.

8.3.1 Performance Requirements

1. **80% of the students should be able to intuitively use the application on the desktop browsers Google Chrome, Safari, Edge and Firefox without further explanation.**

   During the user tests it seemed that most students were quite able to find their way around the system. Most of the situations where they got stuck had to with parts of the system being broken. These problems have been resolved, so we expect this requirement to be fulfilled.

2. **80% of the students should be able to intuitively use the application on the mobile browsers Google Chrome and Safari without further explanation.**

   During the tests students mostly used desktop browsers, so we do not really have any information on mobile browsers, but we expect approximately the same results.

3. **50% of the teachers should be able to intuitively use the application on the desktop browsers Google Chrome, Safari, Edge and Firefox without further explanation.**

   During our user tests the teachers had quite some trouble with the system. This was partially due to our hi-fi prototype breaking, but also due to the system being a bit more complex to learn. We hope that we fixed some of these issues, by improving the design. However, we also wrote a user manual that should be able to help out in cases where the design is not intuitive enough.

4. **A student should be able to evaluate a learning goal within 5 seconds.**

   Evaluating can be done fairly quickly.

5. **Uploading a *proof-of-learning* assignment should take no longer than 2 seconds.**

   Uploading can be done fairly quickly.

6. **The application should be compatible and function as intended on the IT infrastructure of Lyceum de Grundel.**

   The product should be able to run on the servers used by Grundel with some slight adaptations. The application currently runs on the server of Wouter van der Brink. In consultation with Grundel, we will make agreements on how to continue the hosting.

7. **The system should be defended against the top ten most common web application security vulnerabilities, as determined by OWASP[1].**

   By implementing the measures outlined in subsection 5.1.3, we believe to have sufficiently secured our application against these vulnerabilities

[1] https://owasp.org/www-project-top-ten/

9 Future Work

9.1 Introduction

In this section, we will elaborate on what could be done differently for further design projects of the same kind. In addition, we will touch on possible improvements that could be made to our application to make it more user friendly or to offer more usability for our client.

9.2 Future Support

Our client would like to use our product next academic year in a pilot phase. During this pilot phase, they will try the application out with a single class of students. Because of this, future support of the application is required from our side. For this, we have set up a document where we describe the terms and conditions under which we could further work together with our client on this project.

9.3 Possible Improvements

There are several improvements that could be made to our application.

1. A possible improvement is that the teacher can see exactly how many of his students have handed in their proof-of-learning assignment and an overview of a student's hand in history. This would allow the teacher to see if there is a student who is repeatedly neglecting the hand-ins for the assignments.
2. As of now, students can only hand in textual solutions or a picture of their solutions. An additional option could be to allow hand-in via a PDF solution. This is another option for students and might be easier to navigate through for classes where you have to write a lot.
3. Another improvement would be the possibility of removing your own scale-in. It could happen, of course, that a student or teacher miss clicked when scaling in. Therefore, a way must be available to remove this scale in and scale again.
4. Another improvement is the option to remove files in the database.
5. Finally, a setting could be made available for the teacher to switch between an overview that shows the scaling of the whole class and an overview that shows the scaling per student. This allows the teacher to get a quick overview of how the whole class is performing. Additionally, it allows the teacher to show the class how everyone has scaled while keeping anonymity as it does not show students' names. This feature already exists, however it is implemented quite primitively and does not allow for different settings that influence the type of scores that are displayed.

Bibliography

[1] J. Hattie and G. C. Yates, *Visible learning and the science of how we learn*. Routledge, 2013.

[2] S. Ebbens and S. Ettekoven, *Effectief leren*. Noordhoff Uitgevers, 2013.

[3] ——, *Samenwerkend leren*. Noordhoff Uitgevers, 2005.

[4] J. Hattie and H. Timperley, "The power of feedback," *Review of educational research*, vol. 77, no. 1, pp. 81–112, 2007.

[5] D. Sluijsmans and M. Segers, *Toetsrevolutie: naar een feedbackcultuur in het hoger onderwijs*. Uitgeverij Phronese, 2018.

[6] K. Schwaber and J. Sutherland, "The scrum guide," *Scrum Alliance*, vol. 21, no. 1, Nov. 2020.

[7] M. Rehkopf, *User stories with examples and a template*. [Online]. Available: https://www.atlassian.com/agile/project-management/user-stories.

A

Mockups

Figure A.1: The student logs in. The teacher sign-in screen is very similar.

Figure A.2: The student's home screen. Upcoming deadlines are emphasized.

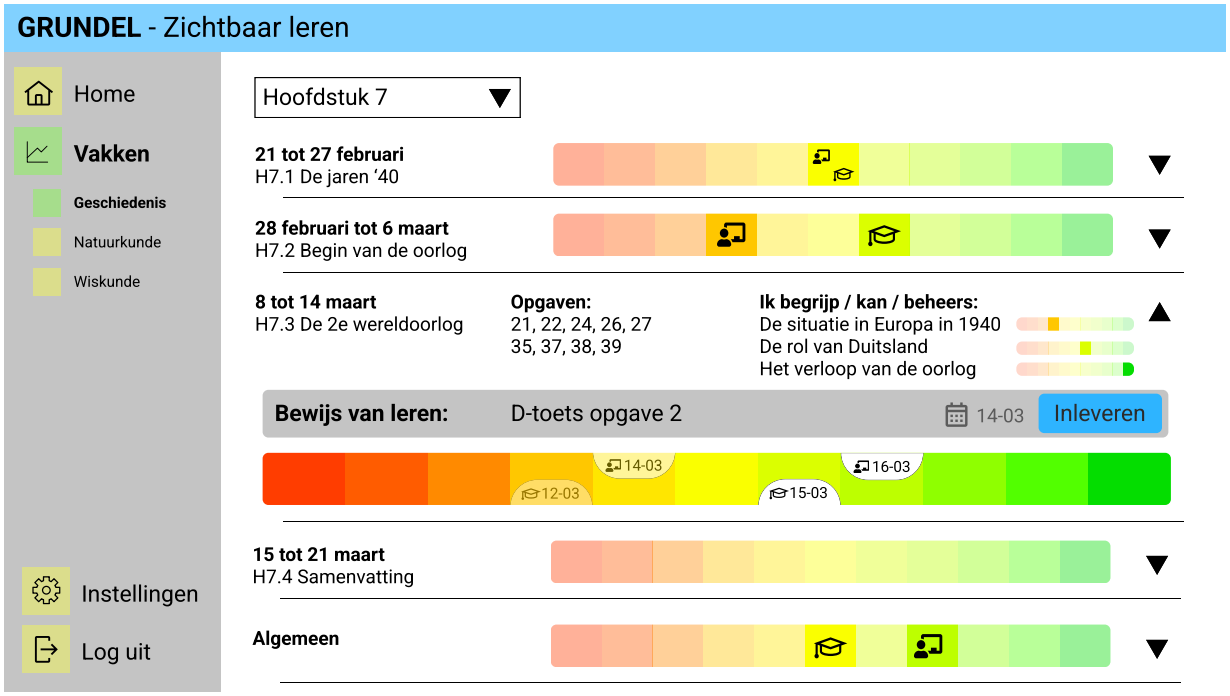


Figure A.3: The student views a chapter.

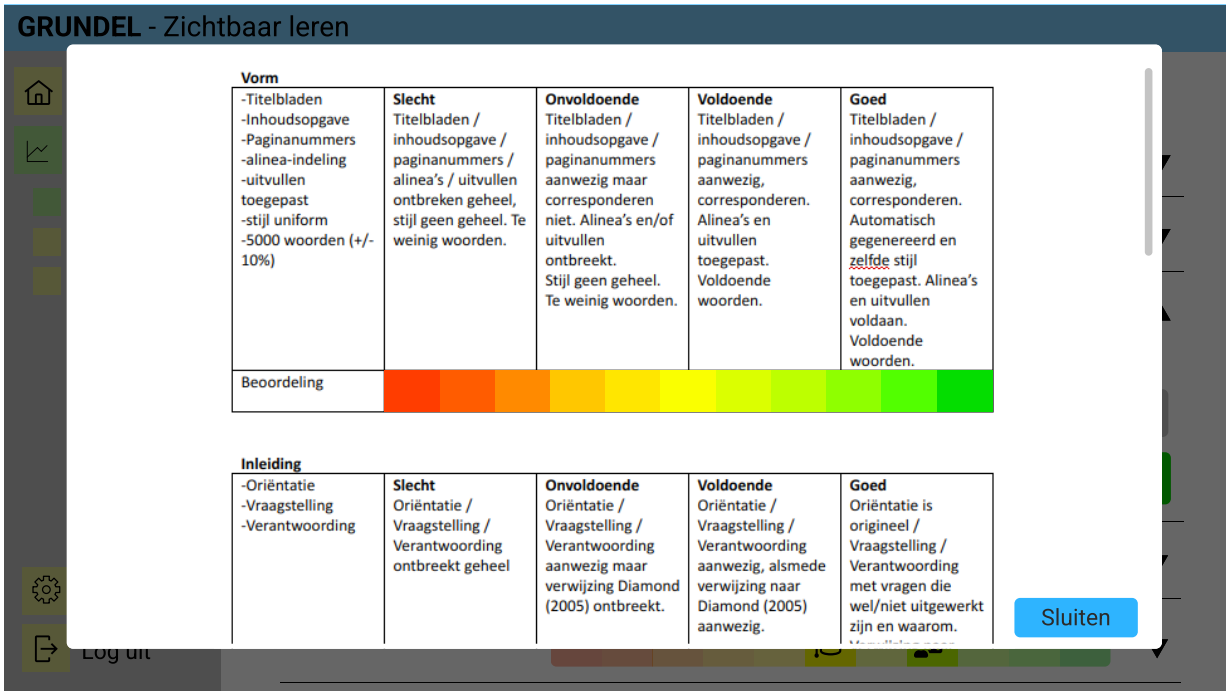


Figure A.4: The student view the rubric for a learning goal.

Figure A.5: The student may submit a proof-of-learning assignment.

Figure A.6: The teacher's home screen.

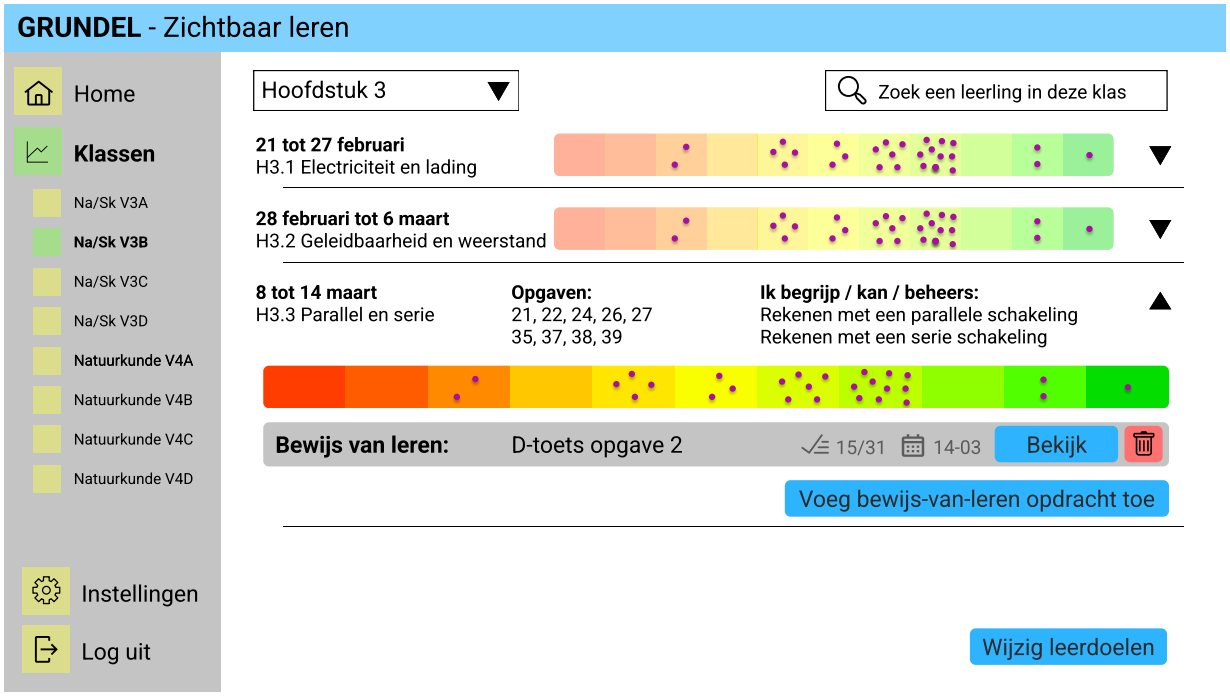


Figure A.7: The teacher views a chapter.

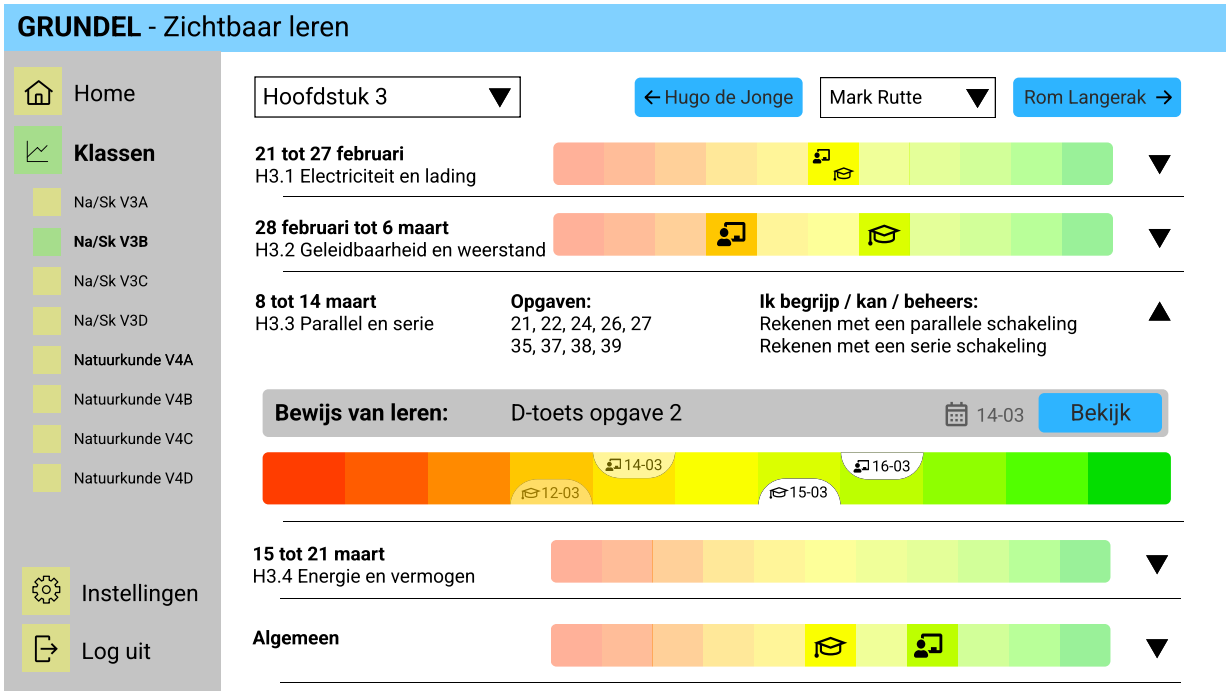


Figure A.8: The teacher views a chapter in the context of a specific student.

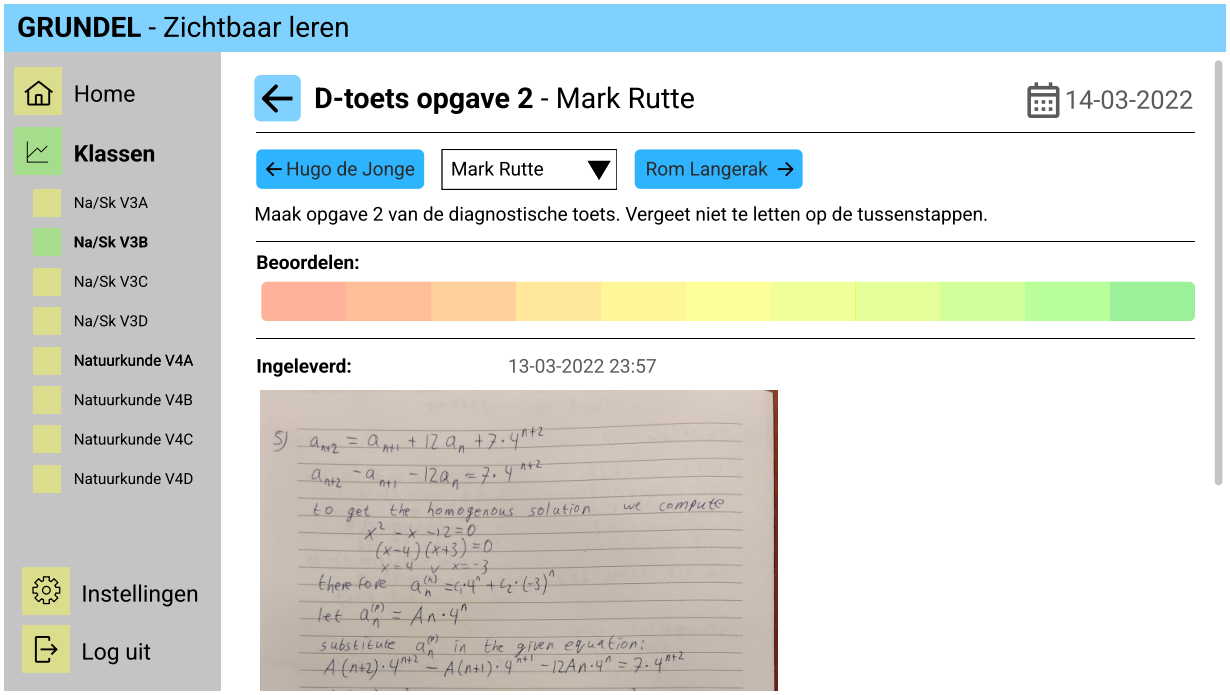


Figure A.9: The teacher reviews an assignment for a specific student.

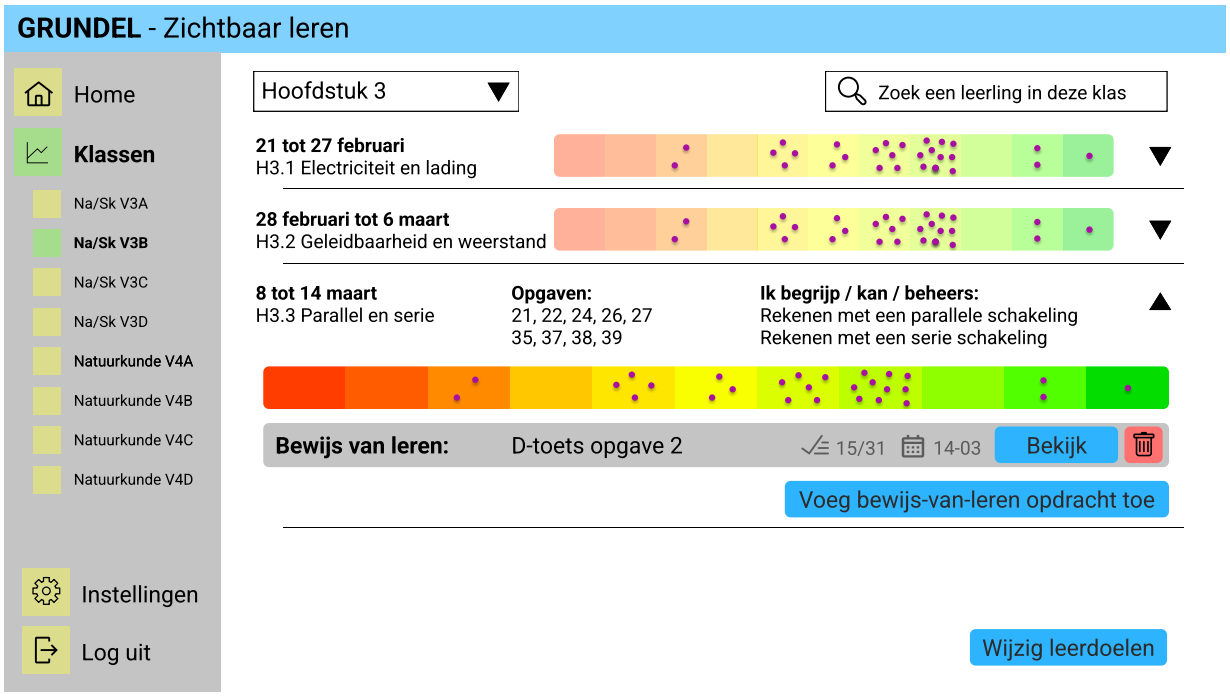


Figure A.10: The teacher reviews an assignment for all students in a group.

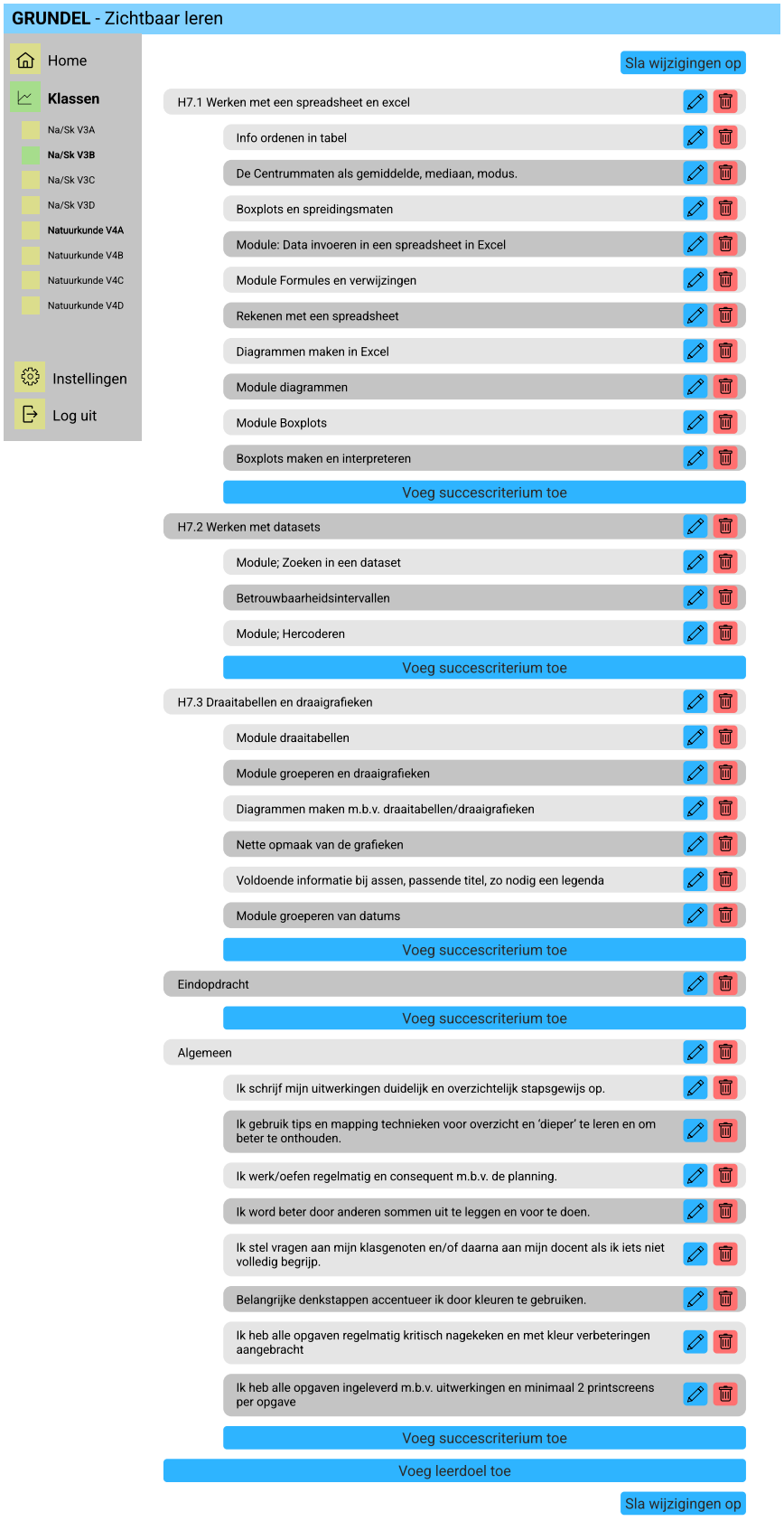


Figure A.11: The teacher edits the learning goals and success criteria for a chapter.

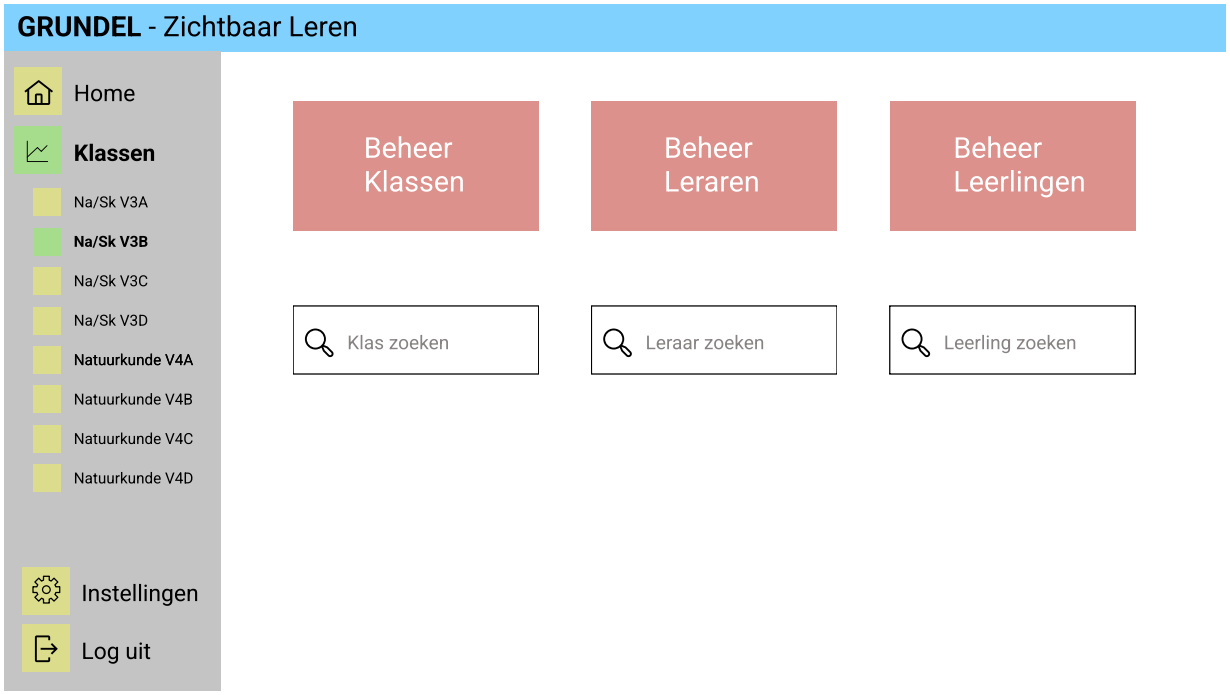


Figure A.12: The adminstrator's home screen.

Figure A.13: The administrator manages the teacher accounts.

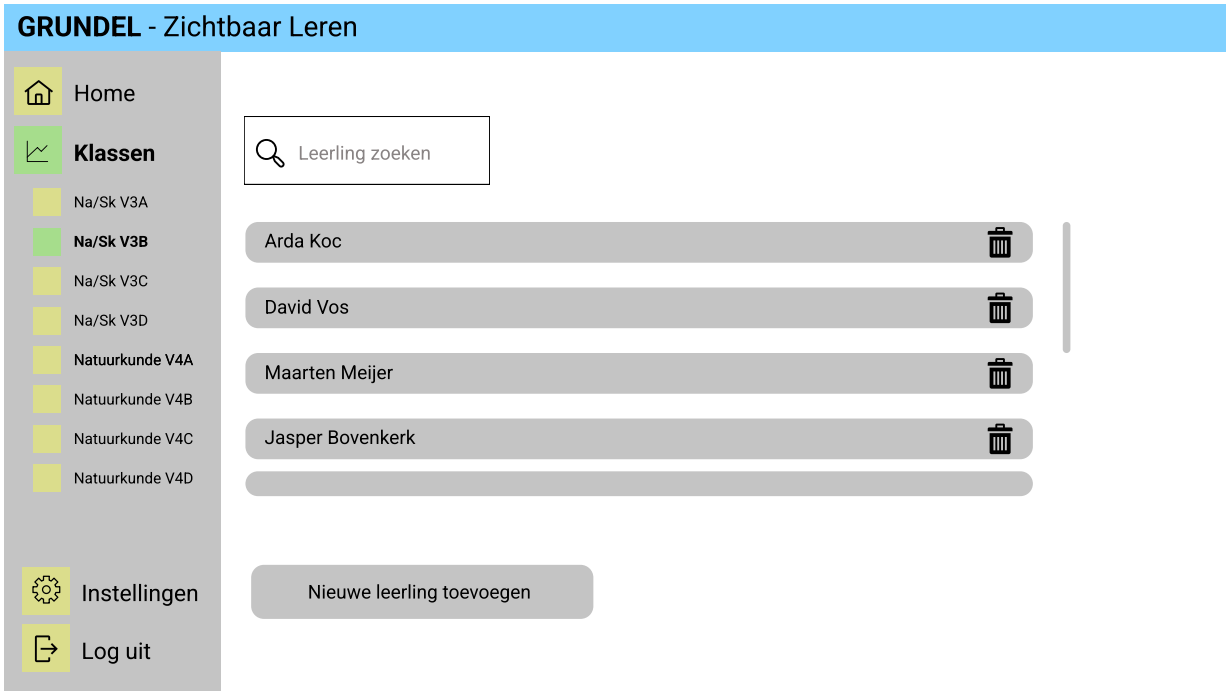


Figure A.14: The administrator manages the student accounts.

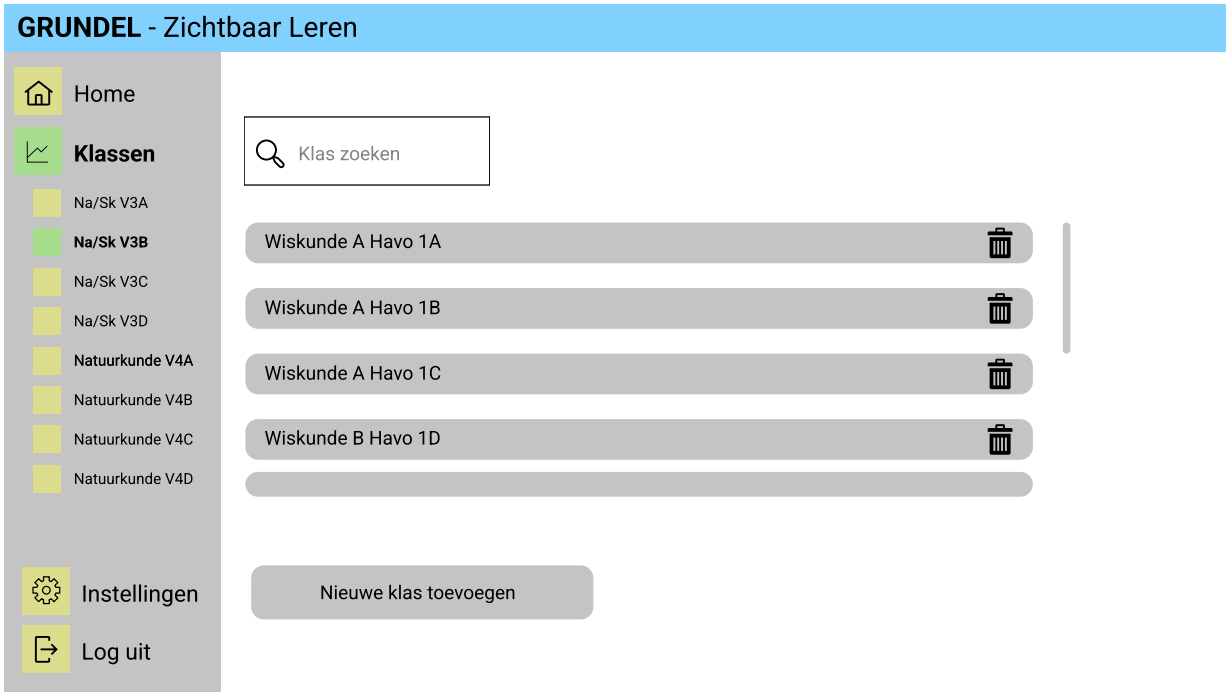


Figure A.15: The administrator manages the classes.

B Test Plan

B.1 User testing 1

B.1.1 Questions teacher

Inleiding

Hartelijk dank dat u mee wilt doen aan de test van ons design project .... Met deze test willen we onderzoeken hoe u het gebruik van het systeem ervaart. Is het systeem bijvoorbeeld eenvoudig in gebruik? En werkt het zoals u verwacht? Of zijn er manieren waarop we de interface kunnen verbeteren? Aan de hand van een stappenplan en vragenlijst wordt u gevraagd gebruik te maken van de verschillende mogelijkheden die de applicatie bieden, en om ons feedback te geven over de verschillende onderdelen.

Over het project

Disco is een applicatie waarmee we proberen leerlingen en docenten te helpen om het leerproces van de leerling zichtbaar te maken. Docenten krijgen de mogelijkheid om de studiewijzer met bijbehorende leerdoelen en succescriteria in te vullen in het systeem. Hierna kunnen leerling en docent zichzelf *inschalen* op een schaal van rood (beheerst het leerdoel nog niet) tot groen (beheerst het leerdoel volledig).

Ook hebben docenten de mogelijkheid om leerlingen formatief te toetsen op de beheersing van het leerdoel door middel van een *bewijs van leren*. Dit is een opdracht die leerlingen zelfstandig dienen te maken en uploaden in het systeem, waarna de docent de opdracht kan voorzien van feedback door middel van inschalen en door het bewijs te voorzien van commentaar.

Uw deelname

Deelname aan de studie is mogelijk als u docent bent in het middelbaar onderwijs. Deelname is geheel vrijwillig en aan deelname zijn geen kosten verbonden. U kunt op ieder moment kiezen om te stoppen.

Tijdens de test verzamelen we verschillende gegevens, waaronder uw persoonsgegevens. We verzamelen de volgende gegevens:

- Voor- en achternaam;
- E-mailadres;
- De vakken of het vak waarvoor u les geeft;
- De antwoorden die u geeft op de gestelde vragen;
- Gegevens die voortvloeien uit het gebruik van het systeem, zoals logbestanden en IP-adressen;

Deze gegevens zijn nodig om de studie uit te voeren. De gegevens worden vertrouwelijk behandeld en zijn toegankelijk voor de auteurs van het project en hun leidinggevende. Uiterlijk twee weken na afloop van het design project, op 2 mei 2022, worden de gegevens permanent vernietigd.

Zoals altijd geeft de AVG u bepaalde rechten, zoals inzage en rectificatie. Wilt u gebruik maken van deze rechten, neem dan contact op en we zullen u zo snel mogelijk helpen.

Stappenplan en vragenlijst

Probeer alle stappen in dit stappenplan te doorlopen. Lukt het niet een bepaalde opdracht uit te voeren, geef dit dan aan en probeer uit te leggen wat er verkeerd ging. Zo kunnen wij proberen de interface te verbeteren, zodat het in het vervolg eenvoudiger wordt om het systeem te gebruiken. Zorgt dit ervoor dat u niet verder kunt met testen, vraag dan om hulp.

Stap 1. Navigeer naar de website `https://lzm.dutch.nl/`.

Stap 2. Log in als docent. Gebruik hiervoor de volgende inloggegevens:

E-mailadres	`j.smit@carmelhengelo.nl`
Wachtwoord	`welkom123`

Stap 3. Ga naar de klas "wiskunde vwo 4".

Stap 4. Ga naar het hoofdstuk "Hoofdstuk 1".

Vraag 1
Hoe eenvoudig was het om het gevraagde hoofdstuk te openen?

Stap 5. Er is een bewijs-van-leren opdracht gemaakt voor hoofdstuk 1.1. Bekijk deze opdracht.

Stap 6. David Vos heeft de opdracht ingeleverd. Bekijk zijn uitwerking.

Stap 7. Schaal de uitwerking van David in.

Stap 8. Bekijk de voorbeeld-uitwerking van deze opdracht.

Vraag 2
Hoe eenvoudig was het om de opdracht te vinden?

Vraag 3
Hoe ervaart u het beoordelen van bewijs-van-leren opdrachten? Gaat het snel genoeg om een klas van 30 leerlingen te kunnen beoordelen?

Vraag 4
Zou u overwegen om dit systeem te gebruiken voor zulke formatieve opdrachten in plaats van bijvoorbeeld een schrift, Magister of SOM? Waarom wel of niet?

Stap 9. Bekijk het leerdoel "Hoofdstuk 1.1 - Exponentiële functies". Er is een bewijs-van-leren opdracht ingeleverd door David. Voorzie zijn opdracht van feedback door in te schalen en commentaar te

leveren.

Vraag 5
Ervaart u de interface voor het beoordelen van deze opdrachten als gebruiksvriendelijk? Waarom wel of niet?

Stap 10. Ga terug naar hoofstuk 1.

Stap 11. Bekijk de voortgang van David Vos.

Stap 12. Schaal David Vos in voor het leerdoel "Hoofdstuk 1.2 - Bewijzen".

Vraag 6
Hoe eenvoudig was het om de leerling te vinden?

Vraag 7
Hoe ervaart u het inschalen van leerdoelen? Gaat het snel genoeg om een klas van 30 leerlingen te kunnen inschalen?

Stap 13. Probeer nu te kijken naar hoe de verschillende pagina's van de applicatie eruit zien.

Vraag 8
Wat vindt u van de user interface? Dat wil zeggen, hoe ervaart u de pagina-indelingen en de manier waarop de pagina's eruit zien?

Vraag 9
Wat vindt u goed aan de interface?

Vraag 10
Heeft u verbeterpunten voor de interface?

Uw gegevens

Ten slotte zouden we graag de volgende gegevens van u verzamelen. De gegevens helpen ons om een beter beeld te krijgen van u als docent. Zoals in de inleiding vermeld gaan we vertrouwelijk met uw gegevens om en worden ze uiterlijk twee weken na afgloop van het design project, op 2 mei 2022, vernietigd.

Voornaam
Achternaam
E-mailadres
Mogen we u benaderen met vervolgvragen naar aanleiding van het onderzoek? Ja / nee

B.1.2 Questions student

Inleiding

Hartelijk dank dat je mee wilt doen aan de test van ons design project .... Met deze test willen we onderzoeken hoe jij het gebruik van het systeem ervaart. Is het systeem bijvoorbeeld eenvoudig in gebruik? En werkt het zoals je zou verwachten? Of zijn er manieren waarop we de interface kunnen verbeteren? Aan de hand van een stappenplan en vragenlijst word je gevraagd gebruik te maken van de verschillende mogelijkheden die de applicatie biedt, en om ons feedback te geven over de verschillende onderdelen.

Over het project

Disco is een applicatie waarmee we proberen leerlingen en docenten te helpen om het leerproces van de leerling zichtbaar te maken. Docenten krijgen de mogelijkheid om de studiewijzer met bijbehorende leerdoelen en succescriteria in te vullen in het systeem. Hierna kunnen leerling en docent zichzelf *inschalen* op een schaal van rood (beheerst het leerdoel nog niet) tot groen (beheerst het leerdoel volledig).

Ook hebben docenten de mogelijkheid om leerlingen formatief te toetsen op de beheersing van het leerdoel door middel van een *bewijs van leren*. Dit is een opdracht die leerlingen zelfstandig dienen te maken en uploaden in het systeem, waarna de docent de opdracht kan voorzien van feedback door middel van inschalen en door het bewijs te voorzien van commentaar.

Je deelname

Deelname aan de studie is mogelijk als je een leerling bent in het middelbaar onderwijs. Deelname is geheel vrijwillig en aan deelname zijn geen kosten verbonden. Je kunt op ieder moment kiezen om te stoppen.

Tijdens de test verzamelen we verschillende gegevens, waaronder je persoonsgegevens. We verzamelen de volgende gegevens:

- Voor- en achternaam;
- E-mailadres;
- De antwoorden die je geeft op de gestelde vragen;
- Gegevens die voortvloeien uit het gebruik van het systeem, zoals logbestanden en IP-adressen;

Deze gegevens zijn nodig om de studie uit te voeren. De gegevens worden vertrouwelijk behandeld en zijn toegankelijk voor de auteurs van het project en hun leidinggevende. Uiterlijk twee weken na afloop van het designproject, op 2 mei 2022, worden de gegevens permanent vernietigd.

Zoals altijd geeft de AVG je bepaalde rechten, zoals inzage en rectificatie. Wil je gebruik maken van deze rechten, neem dan contact met ons op en we zullen je zo snel mogelijk helpen.

Stappenplan en vragenlijst

Probeer alle stappen in dit stappenplan te doorlopen. Lukt het niet een bepaalde opdracht uit te voeren, geef dit dan aan en probeer uit te leggen wat er verkeerd ging. Zo kunnen wij proberen de interface te verbeteren, zodat het in het vervolg eenvoudiger wordt om het systeem te gebruiken. Zorgt dit ervoor dat je niet verder kunt met testen, vraag dan om hulp.

Stap 1. Navigeer naar de website `https://lzm.dutch1.nl/`.

Stap 2. Log in als leerling. Gebruik hiervoor de volgende inloggegevens:

E-mailadres	`12345@mijncarmelhengelo.nl`
Wachtwoord	`welkom123`

Stap 3. Ga naar het vak "wiskunde".

Stap 4. Bekijk hoe de docent je heeft ingeschaald voor het leerdoel "Exponentiële functies" van Hoofdstuk 1.

Vraag 11
Is het duidelijk hoe de docent je heeft ingeschaald? Waarom wel of niet?

Stap 5. Ga naar het deadline overzicht en bekijk de opdracht die je nog moet inleveren.

Stap 6. Upload je antwoord voor deze opdracht door middel van het uploaden van een foto.

Stap 7. Lever je antwoord in.

Vraag 12
Hoe vond je het om zo je antwoorden in te kunnen leveren?

Stap 8. Kijk even rond naar hoe de applicatie eruit ziet en beantwoord de volgende vraag.

Vraag 13
Wat vind je van de user interface? Dat wil zeggen, hoe ervaar je de pagina-indelingen en de manier waarop de pagina's eruit zien?

Vraag 14
Wat vind je goed aan de interface?

Vraag 15
Heb je verbeterpunten voor de interface?

B.2 User testing 2

B.2.1 Questions teacher

Inleiding

Hartelijk dank dat u mee wilt doen aan de test van ons design project. Met deze test willen we onderzoeken hoe u het gebruik van het systeem ervaart. Is het systeem bijvoorbeeld eenvoudig in gebruik? En werkt het zoals u verwacht? Of zijn er manieren waarop we de interface kunnen verbeteren? Aan de hand van een stappenplan en vragenlijst wordt u gevraagd gebruik te maken van de verschillende mogelijkheden die de applicatie bieden, en om ons feedback te geven over de verschillende onderdelen.

Over het project

Disco is een applicatie waarmee we proberen leerlingen en docenten te helpen om het leerproces van de leerling zichtbaar te maken. Docenten krijgen de mogelijkheid om de studiewijzer met bijbehorende leerdoelen en succescriteria in te vullen in het systeem. Hierna kunnen leerling en docent zichzelf inschalen op een schaal van rood (beheerst het leerdoel nog niet) tot groen (beheerst het leerdoel volledig).

Ook hebben docenten de mogelijkheid om leerlingen formatief te toetsen op de beheersing van het leerdoel door middel van een bewijs van leren. Dit is een opdracht die leerlingen zelfstandig dienen te maken en uploaden in het systeem, waarna de docent de opdracht kan voorzien van feedback door middel van inschalen en door het bewijs te voorzien van commentaar.

Uw deelname

Deelname aan de studie is mogelijk als u docent bent in het middelbaar onderwijs. Deelname is geheel vrijwillig en aan deelname zijn geen kosten verbonden. U kunt op ieder moment kiezen om te stoppen.

Tijdens de test verzamelen we verschillende gegevens, waaronder uw persoonsgegevens. We verzamelen de volgende gegevens:

- Voor- en achternaam;

- E-mailadres;
- De vakken of het vak waarvoor u les geeft
- De antwoorden die je geeft op de gestelde vragen;
- Gegevens die voortvloeien uit het gebruik van het systeem, zoals logbestanden en IP-adressen;

Deze gegevens zijn nodig om de studie uit te voeren. De gegevens worden vertrouwelijk behandeld en zijn toegankelijk voor de auteurs van het project en hun leidinggevende. Uiterlijk twee weken na afloop van het design project, op 2 mei 2022, worden de gegevens permanent vernietigd.

Zoals altijd geeft de AVG u bepaalde rechten, zoals inzage en rectificatie. Wilt u gebruik maken van deze rechten, neem dan contact op en we zullen u zo snel mogelijk helpen.

Stappenplan en vragenlijst

Probeer alle stappen in dit stappenplan te doorlopen. Lukt het niet een bepaalde opdracht uit te voeren, geef dit dan aan en probeer uit te leggen wat er verkeerd ging. Zo kunnen wij proberen de interface te verbeteren, zodat het in het vervolg eenvoudiger wordt om het systeem te gebruiken. Zorgt dit ervoor dat u niet verder kunt met testen, vraag dan om hulp.

Stap 1. Navigeer naar de website `https://lzm.dutch1.nl/`.

Stap 2. Log in als leerling. Gebruik hiervoor de volgende inloggegevens:

E-mailadres	`j.smit@carmelhengelo.nl`
Wachtwoord	`welkom123`

Stap 3. Ga naar de klas *Wiskunde h3a.*

Stap 4. Ga naar het hoofdstuk *Wiskunde 3h hoofdstuk 1*

Vraag 16
Hoe eenvoudig was het om het gevraagde hoofdstuk te openen?

Stap 5. Probeer nu naar hoofdstukken overzicht te gaan.

Stap 6. Wijzig de inhoud van het hoofdstuk *Wiskunde 3h hoofdstuk 1* door een extra leerdoel toe te voegen.

Vraag 17
Hoe ervaart u de manier waarop leerdoelen aan hoofdstukken toegevoegd kunnen worden?

Stap 7. Voeg succescriteria aan het leerdoel dat u net gemaakt heeft. Sla uw wijzigingen op.

Vraag 18
Hoe ervaart u de manier waarop succescriteria aan leerdoelen toegevoegd kunnen worden?

Stap 8. Ga nu weer terug naar de home pagina en voeg een nieuwe klas toe.

Stap 9. Geef deze nieuwe klas de naam *Wiskunde h3b*

Stap 10. Importeer de leerlinggegevens van deze klas d.m.v. een Excel-bestand uit SOM

Stap 11. Koppel een wiskunde hoofdstuk aan deze nieuwe klas

Stap 12. Voeg de klas toe

Vraag 19
Hoe ervaart u deze manier van klassen aanmaken? Wat zijn eventuele verbeterpunten?

Stap 13. Er is een bewijs-van-leren opdracht gemaakt voor wiskunde 3h hoofdstuk 1 genaamd Parabolen. Bekijk deze opdracht.

Stap 14. David Vos heeft de opdracht ingeleverd. Bekijk zijn uitwerking.

Stap 15. Schaal de uitwerking van David in.

Stap 16. Bekijk de voorbeeld-uitwerking van deze opdracht.

Vraag 20
Hoe eenvoudig was het om de opdracht te vinden?

Vraag 21
Hoe ervaart u het beoordelen van bewijs-van-leren opdrachten? Gaat het snel genoeg om een klas van 30 leerlingen te kunnen beoordelen?

Vraag 22
Zou u overwegen om dit systeem te gebruiken voor zulke formatieve opdrachten in plaats van bijvoorbeeld een schrift, Magister of SOM? Waarom wel of niet?

Stap 17. Bekijk de opdracht van *Wiskunde h3a - Exponentiële groei*. Er is een bewijs-van-leren opdracht ingeleverd door David. Voorzie zijn opdracht van feedback door in te schalen en commentaar te leveren.

Vraag 23
Ervaart u de interface voor het beoordelen van deze opdrachten als gebruiksvriendelijk? Waarom wel of niet?

Vraag 24
Hoe ervaart u de mogelijkheid om eventueel extra commentaar te leveren bij een ingeleverde opdracht?

Stap 18. Ga terug naar hoofstuk 1.

Stap 19. Bekijk de voortgang van David Vos.

Stap 20. Schaal David Vos in voor het leerdoel *Hoofdstuk 1.2 - Bewijzen.*

Vraag 25
Hoe eenvoudig was het om de leerling te vinden?

Vraag 26
Hoe ervaart u het inschalen van leerdoelen? Gaat het snel genoeg om een klas van 30 leerlingen te kunnen inschalen?

Stap 21. Probeer nu te kijken naar hoe de verschillende pagina's van de applicatie eruit zien.

Vraag 27
Wat vindt u van de user interface? Dat wil zeggen, hoe ervaart u de pagina-indelingen en de manier waarop de pagina's eruit zien?

Vraag 28
Wat vindt u goed aan de interface?

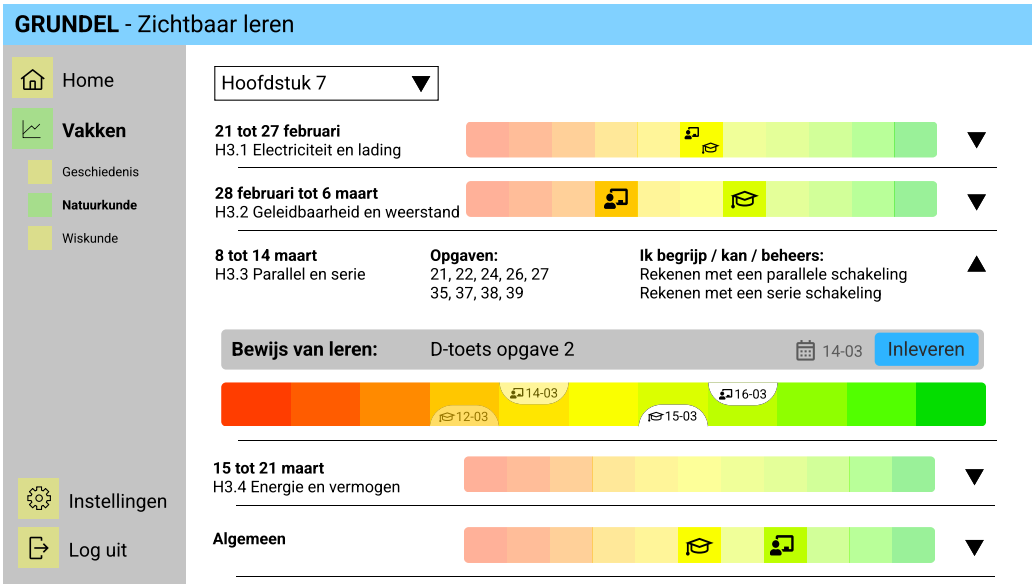
C Planning

Wk.	Start	Tasks	Activities
1	07-02	Create user stores Prioritize user stories	09-02: Introduction, selection of groups and assignments
2	14-02	Write a global description of the product design Describe project approach Create planning	
-	21-02	*Holiday week*	
3	28-02	Create UI scaffolding Create test plan for researchers Create test plan for participants Implement authentication and authorization	01-03: PR: project proposal and planning
4	07-03	Develop scaling interface Implement uploading of proofs of learning Implement retrieving and grading proofs of learning	
5	14-03	Implement class management Implement user management Create administrator overview screen Write reflection essay outline	15-03: PR: requirements and test plan
6	21-03	Implement learning goals Implement success criteria Create student overview screen Create teacher overview screen Work on reflection essay Perform user testing	
7	28-03	Create design report outline Work on design report Evaluate and implement feedback from test results Work on reflection essay	29-03: PR: design and first prototype
8	04-04	Evaluate and implement client feedback Implement remaining features Create presentation outline Create presentation slides Finish and submit reflection essay	

9	11-04	Create poster Prepare demo Write teacher manual Write student manual Write administrator manual	12-04: Presentation 13-04: Deadline reflection essay
10	18-04	Finish design report	20-04: Final poster session

University of Twente

Design Project



Leerlingen- en docentenhandleiding

Group 14

J. Bovenkerk
W. van den Brink
A. Koç
M. Meijer
D. Vos

Supervisor

M. Huisman

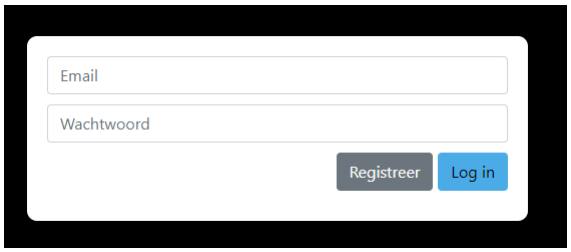
22 april 2022

Inhoudsopgave

1 Handleiding Leerling

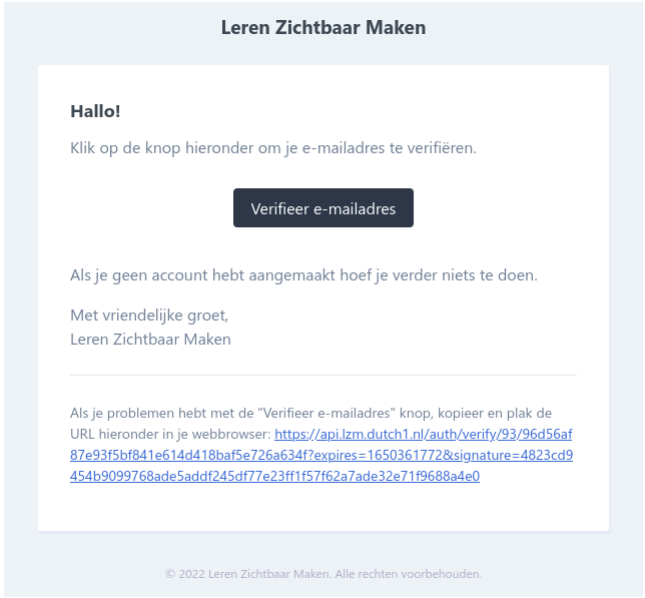
1.1 Inloggen

Ga naar `lzm.dutch1.nl`, je komt nu op de login pagina. Hier kun je inloggen met je account, als je die al eerder gemaakt hebt, of een nieuw account aanmaken door te klikken op *Registreer*.



1.2 Registreren

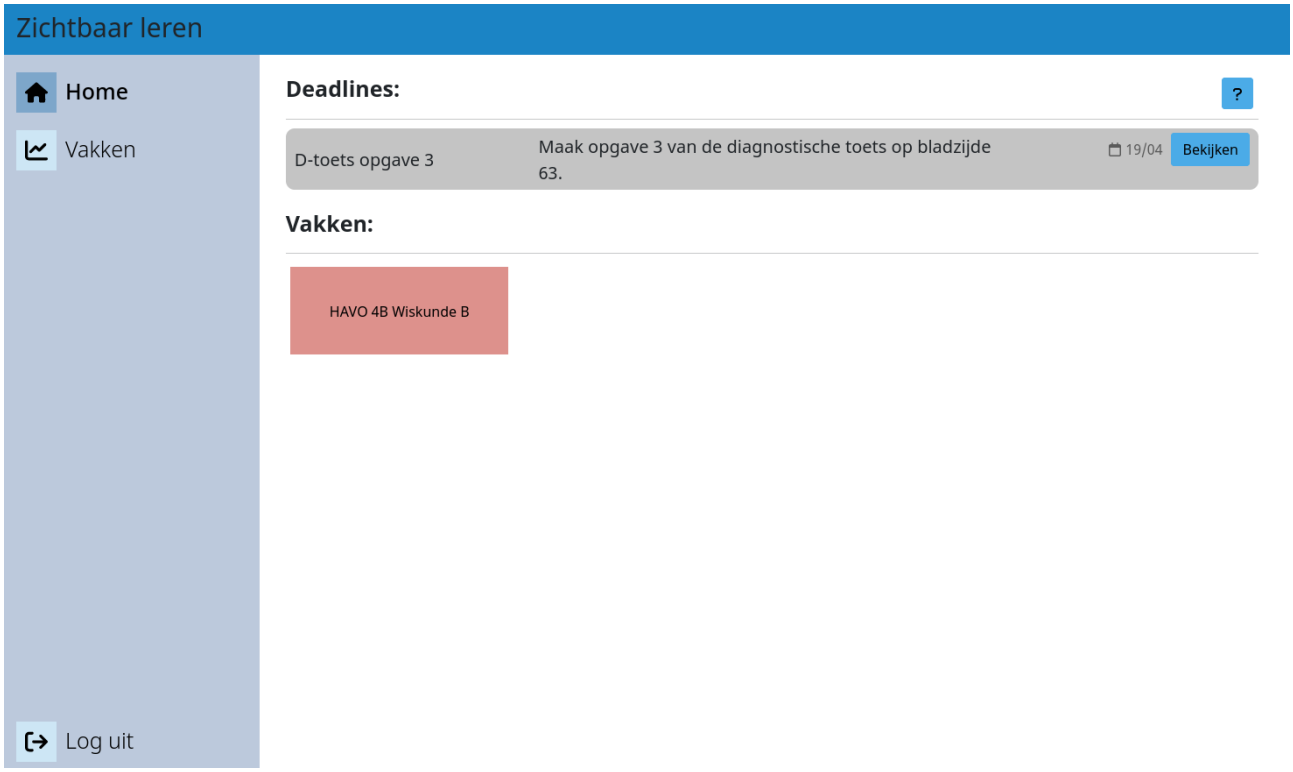
Om te registreren vul je je naam, je Carmel Hengelo email (eindigend op `@mijncarmelhengelo.nl`) en een wachtwoord in. Nadat je op *Registreer* hebt gedrukt ontvang je een email met daarin een knop om je account te bevestigen. Dat ziet er als volgt uit:



Nadat je op de knop *Verifieer emailadres* hebt geklikt kun je inloggen, door je email en wachtwoord in te vullen en op *Log in* te klikken.

1.3 Home

Nadat je ingelogd bent kom je op de homepagina. Deze ziet er zo uit:

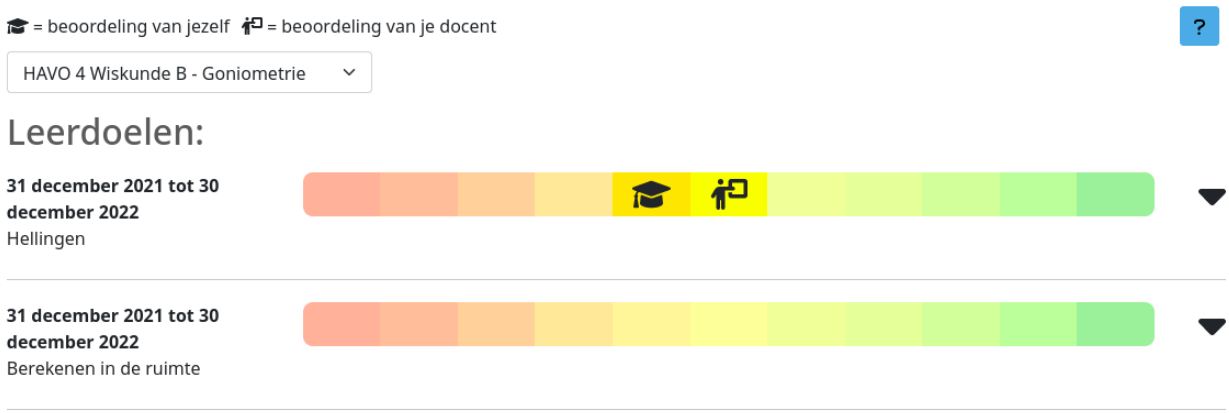


Bovenin zie je als eerste de deadlines van de komende bewijzen van leren die je in kunt leveren. Je kunt snel naar de opdracht toe door op de blauwe knop met *Inleveren* te klikken.

Onder de deadlines zie je een lijst met alle vakken die je volgt. Door op één van de vakken te klikken ga je er naartoe.

1.4 Vak

Nadat je naar een vak gegaan bent, kom je op dit scherm:

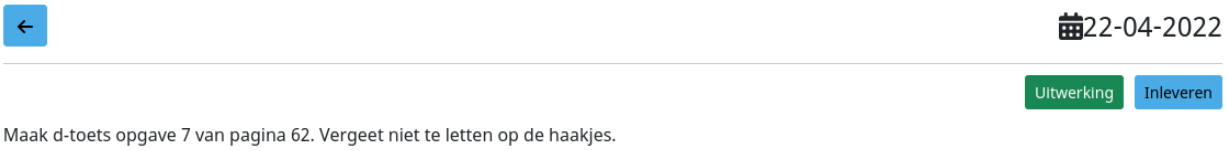


Links bovenin kun je het hoofdstuk kiezen wat je wil bekijken. Vervolgens zie je alle leerdoelen van dat hoofdstuk, dat zijn de verschillende onderwerpen met de gekleurde balkjes ernaast. Je ziet in de gekleurde balkjes de laatste

inschaling die je docent en jijzelf aan je hebben gegeven. Als je op één van de leerdoelen klikt klapt deze uit en zie je meer informatie, bijvoorbeeld zoals dit:

Je ziet hier bijvoorbeeld de verschillende succescriteria en/of de opdrachten die je kunt maken om dit leerdoel te oefenen. Ook kun je hier jezelf inschalen door te klikken op één van de gekleurde balkjes. Rood betekent hier dat je het leerdoel helemaal niet beheerst en groen dat je het leerdoel perfect beheerst. Je docent kan je hier ook inschalen, en je kunt jezelf meerdere keren inschalen zodat je je verbetering in elk onderwerp kunt zien.

Ook kun je, als je docent die heeft toegevoegd, een bewijs van leren maken en inleveren. Als je op de knop *Inleveren* klikt krijg je meer informatie over de opdracht, zoals je hieronder kunt zien:



Vervolgens kun je daar rechtsboven op *Inleveren* drukken en vervolgens een foto of wat tekst inleveren. Ook kun je op de knop *Uitwerking* drukken om de uitwerking te bekijken die je docent heeft geüpload, zodat je jezelf kunt nakijken.

Na het inleveren verschijnt hier ook je ingeleverde opdracht met een balk om jezelf te beoordelen. Ook zie je de feedback die je docent je geeft hier verschijnen, zoals in het voorbeeld hieronder.

22-04-2022

Uitwerking Inleveren

Maak d-toets opgave 7 van pagina 62. Vergeet niet te letten op de haakjes.

Beoordelen:

19-04 19-04

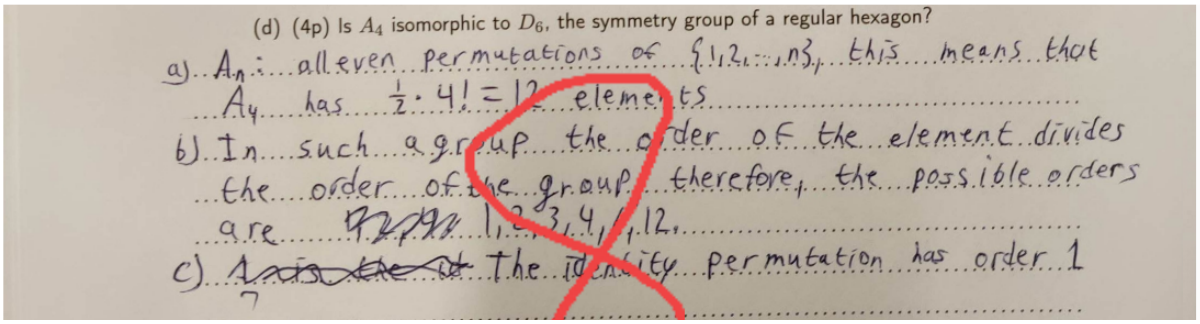
Reacties (1) **Door:** Sara Mercks **Ingeleverd:** 19-04-2022 15:53

Vergeet niet te letten op de kwadraten, ga verder zo door!
19-04-2022 15:55

Nieuwe reactie...

Verzenden

Uitwerking


(d) (4p) Is A_4 isomorphic to D_6, the symmetry group of a regular hexagon?

a) A_n: all even permutations of $\{1,2,\ldots,n\}$, this means that A_4 has $\frac{1}{2} \cdot 4! = 12$ elements

b) In such a group the order of the element divides the order of the group, therefore, the possible orders are 1, 2, 3, 4, 6, 12.

c) The identity permutation has order 1

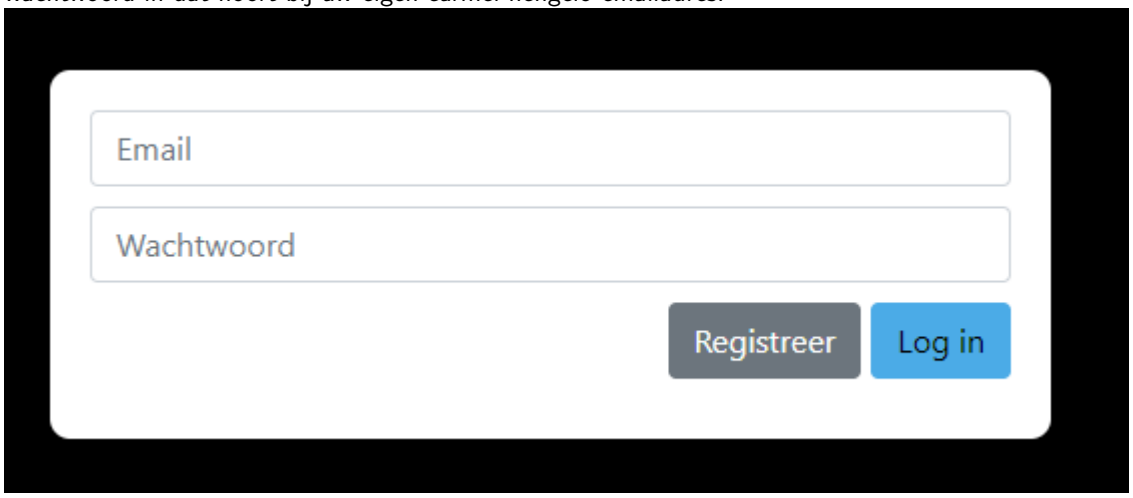
2 Handleiding Leraar

2.1 Inloggen

Navigeer eerst naar de volgende website: https://lzm.dutch1.nl.

2.1.1 Hoe log ik in?

U ziet hier het inlogscherm. Bij email vult u uw eigen carmel hengelo emailadres in. Bij wachtwoord voert u het wachtwoord in dat hoort bij uw eigen carmel hengelo emailadres.



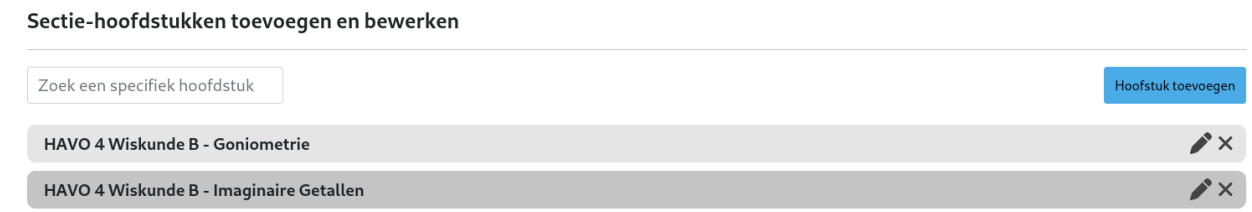
2.2 Hoofdstukken

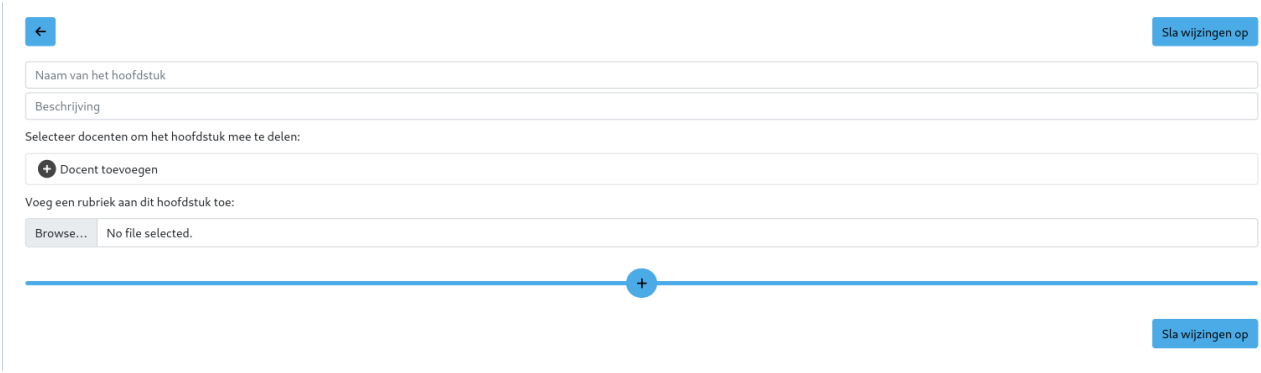
2.2.1 Hoe werken hoofdstukken

In het zijvenster staat de knop *Hoofdstukken*. Dit overzicht duidt op sectie-hoofdstukken, oftewel hoofdstukken die nog niet verbonden zijn aan een specifieke klas. Deze hoofdstukken kan je delen met collega's, en zijn dus handig om te gebruiken met bijvoorbeeld de hele vakgroep of sectie. Deze hoofdstukken kan je zien als studiewijzers of sjablonen, die je later kan koppelen aan een klas. Als je deze hoofdstukken bewerkt, zijn ze ook gewijzigd bij sectiegenoten waar de hoofdstukken mee zijn gedeeld. Dat betekent dat een sectiegenoot een door jou gewijzigd hoofdstuk kan gebruiken en toevoegen aan een klas. Bij klassen staan ook hoofdstukken, maar deze zijn anders. Deze hoofdstukken zijn verbonden aan een klas en die deel je niet met je sectie. Als je deze hoofdstukken bewerkt, gebeurt dat ook alleen voor die ene klas en voor niemand anders. Andersom geld ook dat een gekoppeld hoofdstuk niet bewerkt kan worden door bijvoorbeeld sectiegenoten.

2.2.2 Sectie-hoofdstukken wijzigen

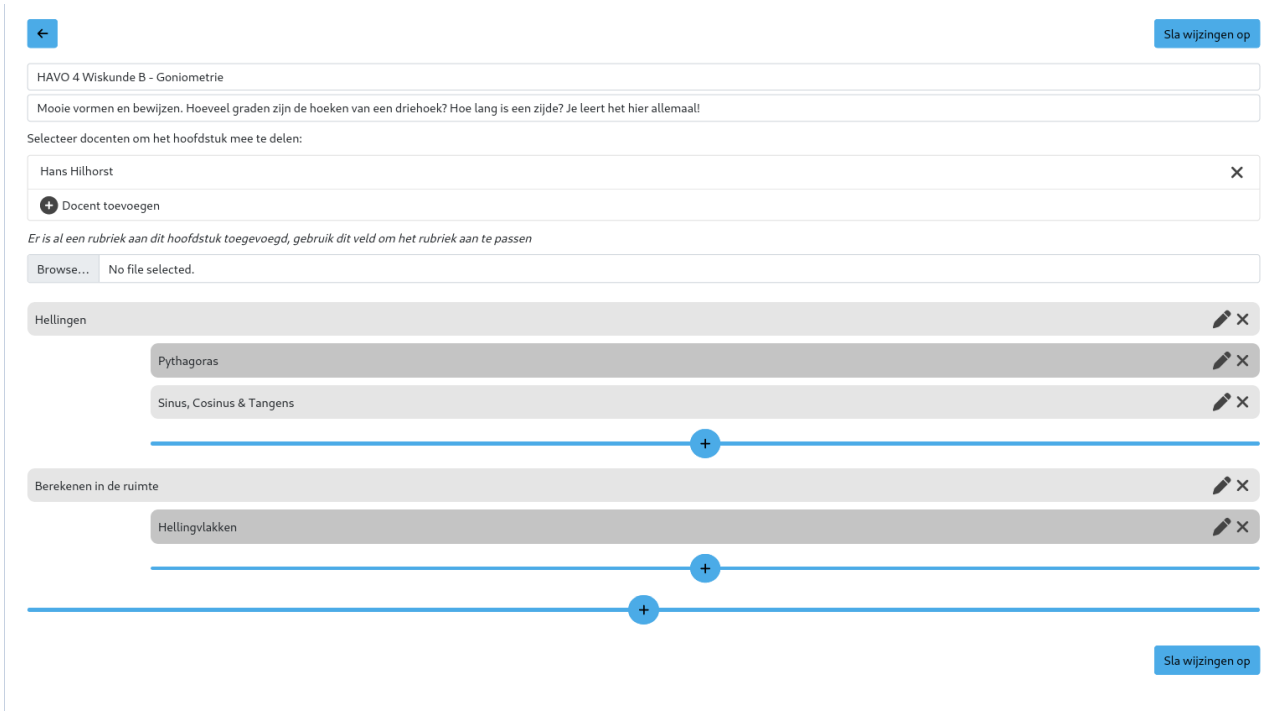
In het overzicht van de knop *Hoofdstukken* kan je nieuwe (sectie-)hoofdstukken aanmaken en bestaande (sectie-)hoofdstukken bewerken. Het aanmaken gaat eenvoudig met de knop *Hoofdstuk toevoegen*, en voor het wijzigen van een hoofdstuk kan je op het pen icoontje klikken bij een hoofdstuk. Beide opties geven een vergelijkbare pagina, namelijk eentje waarbij je alle informatie voor een hoofdstuk kunt toevoegen. Allereerst hebben hoofdstukken altijd een naam. Alle andere opties zijn niet verplicht. Je kunt een beschrijving invullen in het corresponderende veld, en het sectie-hoofdstuk delen met andere docenten. Soms kan het handig zijn om een document toe te voegen aan een hoofdstuk. Hier is ruimte voor gemaakt met de naam *rubriek*. Hiermee kan je een fotobestand toevoegen aan een hoofdstuk. Dit kan een handige plek zijn om een studiewijzer, planning of iets anders toe te voegen wat te maken heeft met het hele hoofdstuk.

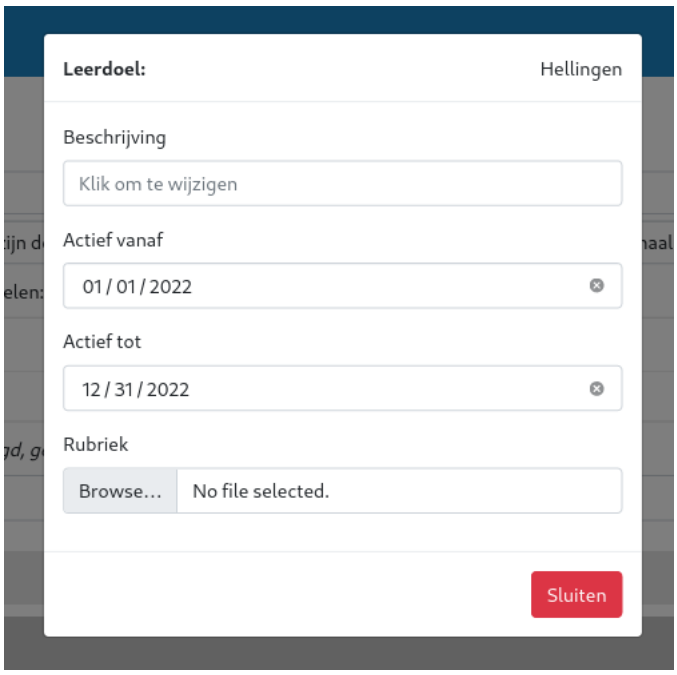




Het toevoegen van leerdoelen en successcriteria kan met het blauwe plus icoontje in de blauwe balk. Als hierop geklikt wordt, verschijnt er vanzelf een leerdoel. De breedte van de blauwe balk onder het plus icoontje geeft aan of het knopje voor een leerdoel of voor een successcriteria is. Leerdoelen zijn groot en successcriteria vallen daaronder, dus die zijn klein. Om de tekst van een leerdoel of successcriterium aan te passen, kan je op de tekst klikken en een gewenste naam intypen. Ook zijn er meer gegevend die je kan aanpassen bij een successcriterium of een leerdoel. Successcriteria hebben alleen een beschrijving, die je kan aanpassen met het pen icoontje. Leerdoelen hebben meer opties, namelijk een beschrijving, een datum vanaf wanneer het leerdoel actief is, een datum tot wanneer het leerdoel actief is en weer een rubriek. Dit rubriek hoort bij een leerdoel, dus hier kan je fotobestanden uploaden die te maken hebben met dat specifieke leerdoel. Wanneer een leerdoel of successcriterium niet op de juiste plaats staat, kan je de volgorde van dezen veranderen. Om dit te doen, kan je een een leerdoel of successcriterium met de muis slepen naar de goede plek.

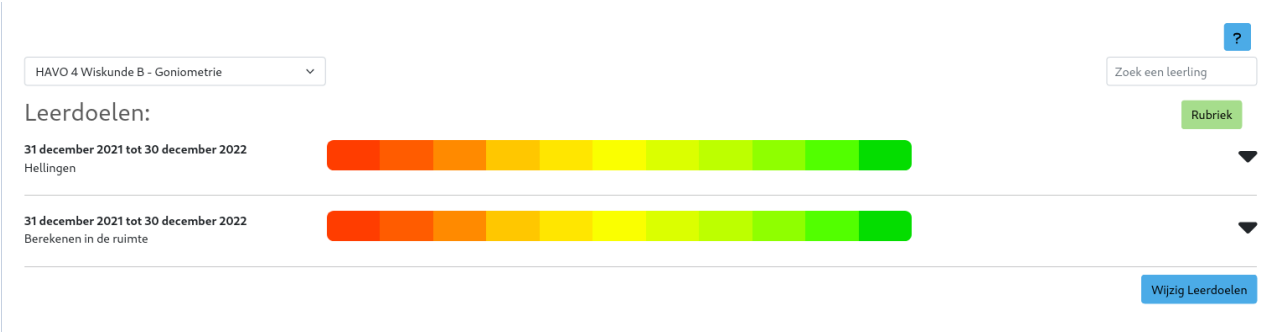
Nadat de gewenste gegevens zijn ingevuld en aangepast, is het belangrijk dat je de wijzigingen opslaat. Hiervoor staan boven- en onderaan de pagina knoppen. Als er wijzigingen zijn aangebracht die niet zijn opgeslagen, zal het systeem een melding geven als je probeert weg te gaan.





2.2.3 Gekoppelde hoofdstukken wijzigen

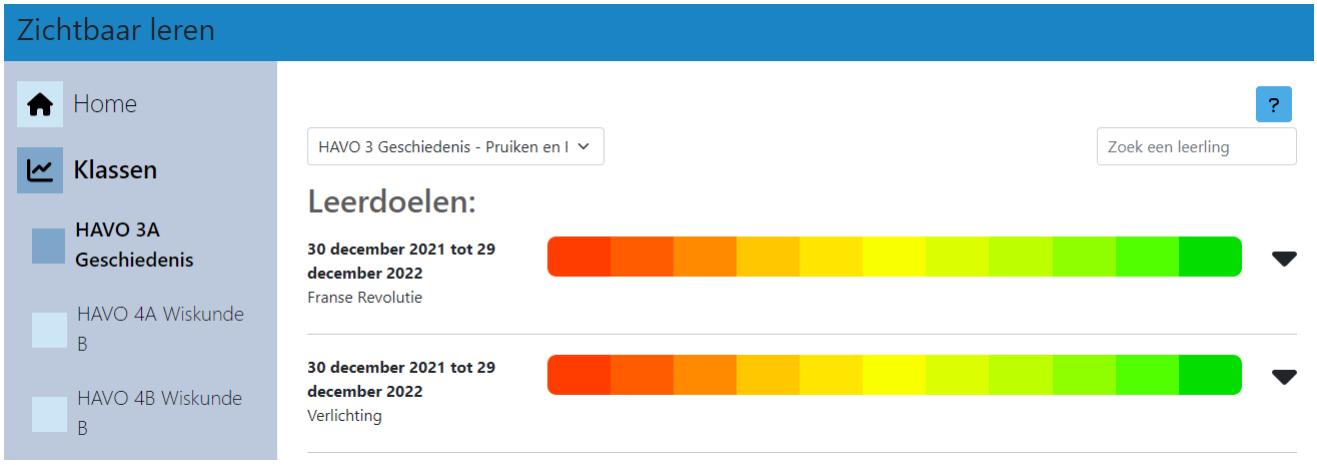
Het wijzigen van hoofdstukken die aan een klas zijn gevoegd werkt heel vergelijkbaar aan het wijzigen van sectie-hoofdstukken. Je kunt de knop om deze hoofdstukken te wijzigen vinden als je in het zijvenster naar *Klassen* gaat, dan een hoofdstuk selecteert en dan op de knop *Wijzig leerdoelen* klikt. Het volgende scherm werkt hetzelfde als eerder beschreven, maar werkt alleen voor die klas.



2.3 Klassen

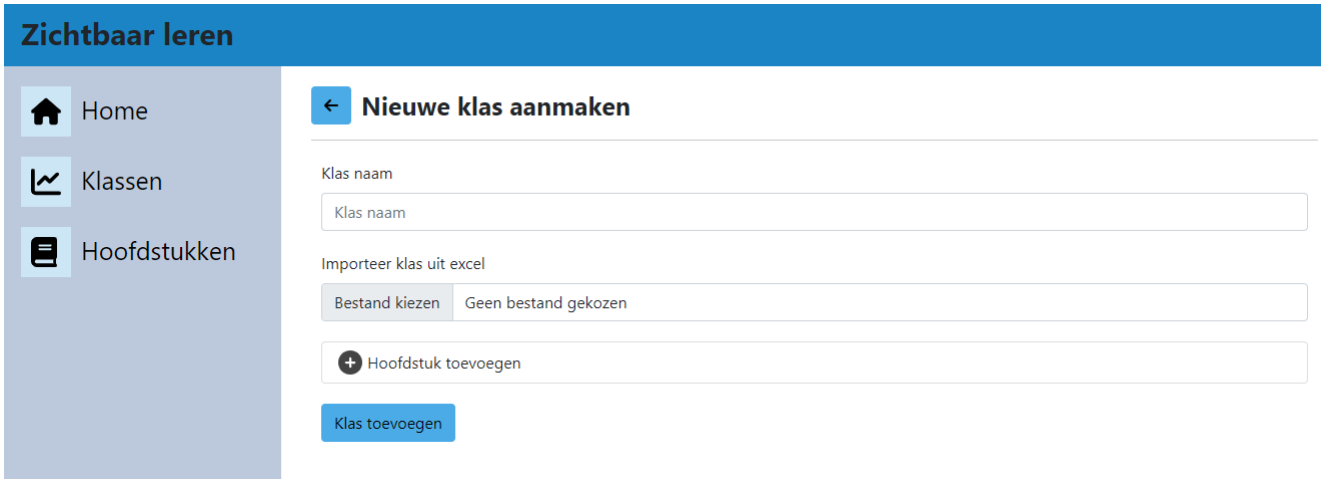
2.3.1 Klassen bekijken

In het zijvenster klik op *Klassen*. Een selectie van uw klassen zal nu verschijnen. Klik op de gewenste klas om de voortgang van deze klas voor een specifiek hoofdstuk te bekijken.



2.3.2 Nieuwe klas toevoegen

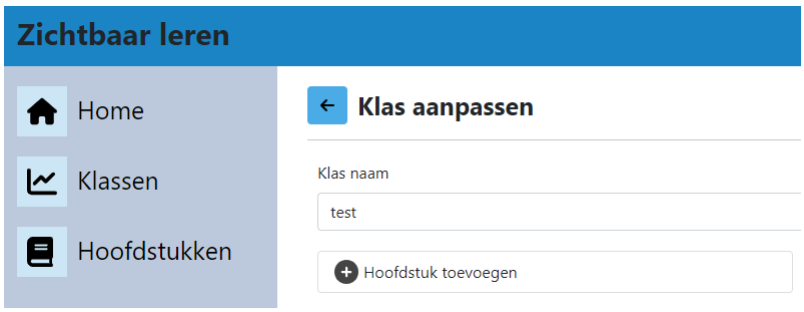
In het zijvenster klik op *Home*. Klik op de knop *Klas toevoegen*. Voeg bij *Klas naam* de naam van de nieuwe klas toe. Om leerlingen toe te voegen aan de klas, kunt u een excel bestand importeren en toevoegen bij *Importeer klas uit excel*. Bij het knopje *Hoofdstuk toevoegen* kunt u de hoofdstukken, die nodig zijn, toevoegen voor deze nieuwe klas. Deze hoofdstukken komen uit het kopje *Hoofdstukken* van het zijvenster. Zodra een hoofdstuk hiervan wordt toegevoegd aan een klas, zal er een kopie worden gemaakt van dat hoofdstuk zoals die op dat moment is. Collega's waarmee het hoofdstuk gedeeld was zullen dus niet het hoofdstuk van je klas kunnen aanpassen, maar alleen jij. Als u alle informatie ingevuld heeft, kunt u de nieuwe klas toevoegen door te klikken op *Klas toevoegen*. Hierna kunt u klikken op het pijltje terug om uit het klas toevoegen scherm te gaan.



2.3.3 Klas wijzigen

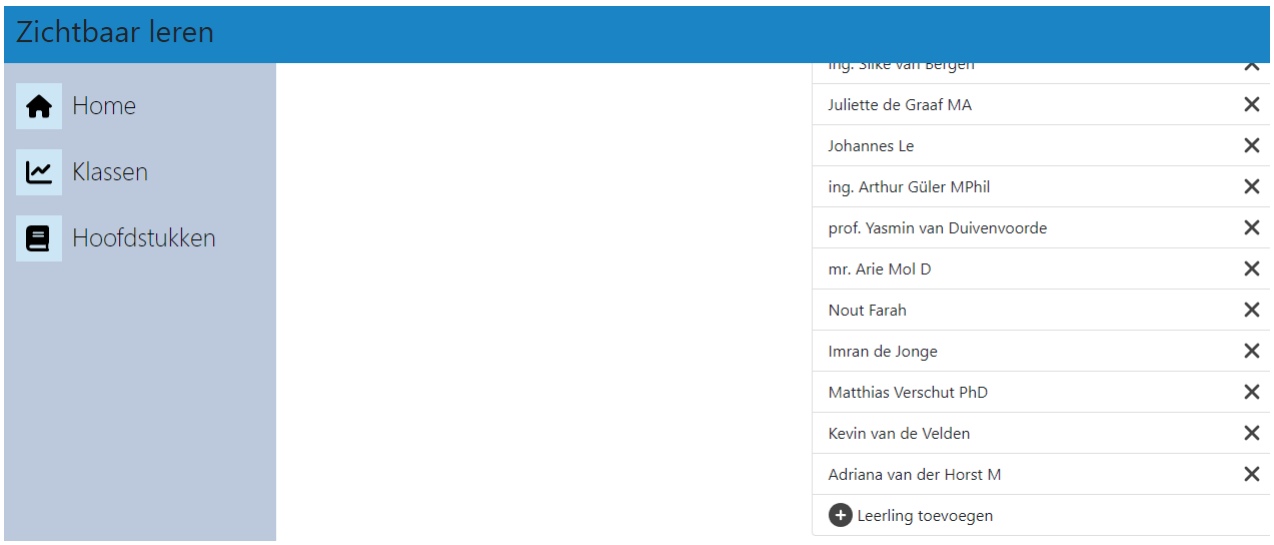
Nieuw hoofdstuk toevoegen

In het zijvenster klik op *Home*. U ziet nu een overzicht van al uw klassen. Klik op het pennetje van de gewenste klas. In dit scherm kunt u de naam aanpassen van de klas onder *Klas naam*. Bij het knopje *Hoofdstuk toevoegen* kunt u (nieuwe) hoofdstukken toevoegen aan een bestaande klas. Uw wijzigingen worden opgeslagen als u op *Aanpassen* klikt. Door op het pijltje terug te klikken, verlaat u het aanpasscherm.



Nieuwe leerling toevoegen

In het zijvenster klik op *Home*. U ziet nu een overzicht van al uw klassen. Klik op het pennetje van de gewenste klas. In dit scherm kunt u de naam aanpassen van de klas onder *Klas naam*. Bij het knopje *Leerling toevoegen* kunt u (nieuwe) leerlingen toevoegen aan een bestaande klas. Uw wijzigingen worden opgeslagen als u op *Aanpassen* klikt. Door op het pijltje terug te klikken, verlaat u het aanpasscherm.



2.4 Bewijs-van-leren opdrachten

2.4.1 Nieuwe opdracht aanmaken

Klik op *Klassen* in het zijvenster. Kies nu de klas waarvoor u een bewijs-van-leren opdracht wilt maken. Navigeer naar het juiste hoofdstuk. U kunt nu op het gewenste leerdoel klikken en een bewijs-van-leren opdracht aanmaken. U doet dit door op het blauwe plus-teken te klikken. Hier kunt u de naam van de opdracht invoeren en de opdracht aanmaken. Elke nieuwe opdracht die u aanmaakt zal ook een inschaal aanmaken.

2.4.2 Inschalen van leerlingen

U heeft bij elk bewijs-van-leren opdracht een inschaal bar waar u uw leerlingen kunt inschalen op hoe goed ze het leerdoel beheersen. Om leerlingen in te schalen klikt u bij de gewenste bewijs-van-leren opdracht op het knopje *Bekijken*. U krijgt nu een overzicht van alle uitwerkingen die de leerlingen hebben ingeleverd bij deze opdracht. Bij elke uitwerking staat een inschaal bar, die gaat van rood naar groen. Kijkend naar hoe goed een leerling het leerdoel beheerst kunt u de leerling inschalen door op een kleur in de inschaal bar te klikken. Dit zal automatisch uw beoordeling opslaan en laten zien aan zowel uzelf als aan de leerling.