

# **Adaptive Study Management Platform and Recommendation using Artificial Intelligence**

Bryan Spahr



<b>Author(s)</b> Bryan Spahr	
<b>Degree programme</b> Bachelor's Degree in Business Information Technology	
<b>Report title</b> Adaptive Study Management Platform and Recommendation using Artificial Intelligence	<b>Number of pages and appendix pages</b> <b>84 + 5</b>
<p><b>Abstract</b></p> <p>This paper has been written as part of my studies in Business Information Technology at the Haaga-Helia University of Applied Sciences. The following research corresponds to my bachelor thesis. The goal of this paper is to present and comment the development of an adaptive study management platform that uses different sub-fields of artificial intelligence.</p> <p>In collaboration and with Dr. Amir Dirin, we decided to resume the original idea and work of Dr. Laine and himself "<i>Towards an Adaptive Study Management Platform: Freedom Through Personalization</i>". We put the theory into practice and developed a solution for both teachers and students regarding study in general.</p> <p>The project we developed is a full stack application composed of several layers. Its architecture is a three-tier architecture that includes a presentation tier, a logic tier and a data tier. The project includes a mobile application for students to study, an adaptive website for teachers to manage their courses, a back-end server with an API and a NoSQL database to save and store the data.</p> <p>The objective of this solution is to provide students a modern and new way to study and teachers an adaptive and proactive way to teach. The end goal is to apply artificial intelligence sub-fields such as data mining and machine learning to provide students a personalized study path recommendation.</p> <p>The following paper covers an introduction part, research questions and methodology, discussions about related research, a design and implementation part, results to some study and test cases alongside with a general discussion and recommendations.</p>	
<p><b>Keywords</b> Adaptive Study Platform, Mobile, Artificial Intelligence, Data mining, Recommendation</p>	

## Acknowledgments

First of all, I would like to thank Mr. Amir Dirin with whom I worked and collaborated to write this thesis. Mr. Dirin was not only my thesis supervisor this semester but is also a lecturer at Haaga-Helia and was my teacher last semester. He is the one who introduced me to the field and concepts of artificial intelligence during his course “Future Learning Research”. Prior to that course, I heard about artificial intelligence, machine learning, data mining and other terms like that but I had no clear idea of what it was exactly and most importantly what were the differences between one and another.

Last semester, I learnt more about artificial intelligence and its sub-fields and it really raised my interest. I was extremely curious about it but I was also very sceptical about my ability to write a research paper about it. It seemed like a very difficult and complex area to work on.

Mr. Dirin shared with me some of his prior researches and gave me four or five different ideas of projects mixing different fields of AI. He then offered me the opportunity to work on one of his own research and to develop what he imagined in collaboration with Dr. Teemu Laine. After reading his paper, I was convinced that I wanted to write my bachelor thesis on that topic and I wanted to develop the platform he imagined. I was extremely enthusiastic about this project.

I am grateful to him for allowing me to pursue his research. I am also very thankful to him for accepting to follow me these last 6 months and help me write my thesis. He gave me leads and ideas that I could exploit, feedback and constructive comments about my work and how I could improve it. He also set up the study case for me allowing me to receive feedback on my work from real test users. I would like to thank Mr. Dirin for his support and assistance from the beginning of the project until now. It would not have been possible without him.

I also would like to express my sincere thanks to the management and the faculty of the Haaga-Helia University of Applied Sciences that accepted me as a student and provided me an idea working and learning environment. I have always received the help and necessary support from the school since the day I arrived until now. I am deeply and extremely grateful to them for the opportunity they gave me and I am truly happy to have spent the last year of my studies there.

Special thanks to Mr. Nicolas Debons, head of the Business Information Technology department at the HES-SO University of Applied Sciences, my home university, for giving me the opportunity to study abroad this year. Thanks to Mrs. Blomster, academic advisor and work placement coordinator at Haaga-Helia who warmly welcomed my colleagues

and me at our arrival but also followed and helped us every time she could. She definitely made us feel at home in Helsinki. Thanks to Mr. Juha Hinkula, software development lecturer at Haaga-Helia who taught me everything I know about mobile and front-end development especially regarding react native and react.js. I wouldn't have been able to develop any of the solutions I built without his lessons.

Finally, I would like to thank all my colleagues from Switzerland and Finland for helping me when I needed and giving me advice. They made the last 12 months even more beautiful and memorable.

## Table of contents

1	Introduction.....	5
2	Research questions and methodology .....	7
2.1	Questions .....	7
2.2	Methodology.....	7
2.2.1	Test case .....	7
2.2.2	Study case .....	10
3	Related research.....	11
4	Design .....	14
4.1	Mobile Application .....	14
4.2	Website .....	26
5	Implementation .....	39
5.1	Mobile application .....	39
5.2	Website .....	40
5.3	Back-end server .....	42
5.4	Database.....	47
5.4.1	NoSQL .....	47
5.4.2	Document-oriented stores .....	47
5.4.3	Collections and documents .....	48
5.5	Global architecture.....	54
5.6	Student data .....	55
5.6.1	Course statistics .....	56
5.6.2	Resource statistics.....	59
5.6.3	Quiz statistics.....	59
5.7	Recommendation.....	60
5.7.1	Data mining.....	61
5.7.2	Machine learning .....	62
6	Result .....	68
6.1	Study case .....	68
6.1.1	Interview results .....	68
6.1.2	User Experience ratings .....	70
6.1.3	Bugs and issues .....	71
6.1.4	Improvements .....	71
6.1.5	Conclusion .....	72
6.2	Test case.....	73
6.2.1	Example 1 .....	73
6.2.2	Example 2 .....	77
6.2.3	Conclusion .....	79

7	Discussion .....	81
8	Recommandation .....	82
9	References .....	84
10	Appendices.....	87
10.1	Report structure .....	87

## List of figures

Figure 1 : Sample students data .....	8
Figure 2 : Student 1 data .....	8
Figure 3 : Student 1 in the database.....	9
Figure 4 : The 3 different study phases based on Dr. Dirin and Laine's research (Dirin & Laine, 2018) .....	11
Figure 5 : Samples of phase 2 prototype screenshots designed by Dr. Dirin and Laine (Dirin & Laine, 2018) .....	14
Figure 6 . Log in screen .....	15
Figure 7 : Sign up screen.....	16
Figure 8 : Dashboard screen .....	17
Figure 9 : Example of a course screen (JavaScript Introduction) .....	18
Figure 10 : The infos tab giving the course's information.....	19
Figure 11 : The achievements screen with results' details.....	20
Figure 12 : Quiz screen with multiple-choice questions .....	21
Figure 13 : A completed quiz with correct answers.....	22
Figure 14 : External website displayed in the current application .....	23
Figure 15 : Screenshot of the home screen .....	24
Figure 16 : Recommendation screen with a ranking of the courses.....	25
Figure 17 : Home page of the website.....	26
Figure 18 : Sign Up page .....	27
Figure 19 : List of all courses dispensed at Haaga-Helia.....	28
Figure 20 : Courses list from an administrator point of view .....	28
Figure 21 : Course details .....	29
Figure 22 : Form to edit a course.....	29
Figure 23 : List of students studying at Haaga-Helia .....	30
Figure 24 : List of teachers teaching at Haaga-Helia .....	31
Figure 25 : Dashboard screen from the teacher point of view .....	31
Figure 26 : Structure and content of the course organized in weeks.....	32
Figure 27 : Creating a quiz.....	32
Figure 28 : Writing the questions and answers of the quiz .....	33
Figure 29 : Adding an external link as a resource.....	34
Figure 30 : Participants of the UX Design course .....	35
Figure 31 : Student course profile.....	36
Figure 32 : Example of student's quiz result.....	37
Figure 33 : Quiz with statistics .....	37
Figure 34 : Full architecture designed by Dr. Dirin and Laine (Dirin & Laine, 2018) .....	39

Figure 35 : Structure of React Native (5 key advantages of React Native, 2017) .....	40
Figure 36 : MVC Architecture (Deutsch, 2017) .....	40
Figure 37 : Sample of render method of the "Students" component .....	41
Figure 38 : Custom React element "Table" .....	42
Figure 39 : Structure of the back-end server .....	43
Figure 40 : Graphic of the MERN Stack components communication (Roy, 2018) .....	44
Figure 41 : User model in the back-end server .....	44
Figure 42 : Object mapping between Node.js and MongoDB via Mongoose .....	45
Figure 43 : Examples of routes from the server side .....	46
Figure 44 : Records in a document-oriented database .....	48
Figure 45 : Database in MongoDB Compass .....	49
Figure 46 : Example of a course document.....	50
Figure 47 : Week document with its arrays .....	51
Figure 48 : Overview of the teachers collection .....	51
Figure 49 : User documents.....	52
Figure 50 : The role of the user is stored in the session storage after log in .....	53
Figure 51 : The role of the user is retrieved from the session storage .....	53
Figure 52 : Simple interaction between the user and the components.....	55
Figure 53 : Sequence diagram illustrating the saving of the timestamp.....	56
Figure 54 : Time spent by a student on a specific course.....	57
Figure 55 : componentDidMount and componentWillUnmount methods of the Course component.....	58
Figure 56 : Examples of lecture's stats.....	59
Figure 57 : Example of a quiz document in the database .....	60
Figure 58 : Example of student data stored in the database .....	62
Figure 59 : Application of the formula .....	64
Figure 60 : Ordering the courses by their total score .....	65
Figure 61 : Sorting of the courses by their study path.....	66
Figure 62 : Sorting the different paths to recommend one.....	67
Figure 63 : Ease of use of the application (from 1 to 7) .....	69
Figure 64 : How likely the test users are going to use the application (1-7) .....	70
Figure 65 : Logged in as Lydia Pace (student 1).....	73
Figure 66 : Lydia's JavaScript Introduction course overview and achievements .....	74
Figure 67 : Time to seconds converter .....	75
Figure 68 : Values for student 1 .....	75
Figure 69 : Recommendation table on student 1 account.....	76
Figure 70 : Students with extremely close values .....	77
Figure 71 : Recommendation for Adam Branch.....	78



Figure 72 : Example of data and results for a single student .....	79
Figure 73 : Results of the test case .....	80

## List of abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
App	Application
Avg	Average
CSS	Cascading Style Sheets
DB	Database
DBMS	Database Management System
HTML	Hypertext Markup Language
ID	Identifier
IoT	Internet of Things
JS	JavaScript
JSON	JavaScript Object Notation
JSX	JavaScript XML
MERN	MongoDB Express.js React.js Node.js
MVC	Model View Controller
NB	Number
NoSQL	Not only SQL
ODM	Object Data Modelling
PDF	Portable Document Format
RDBMS	Relational Database Management System
SQL	Structured Query Language
UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UX	User eXperience
XML	Extensible Markup Language

# 1 Introduction

Nowadays, smartphones are widely used and the further we go in time, the more powerful and convenient they become. While it was only used to make phone calls 20 years ago, smartphones are now used for everything in a daily basis. There are millions of mobile applications available on the market allowing you to do almost everything. You can order food, control your TV or learn new languages simply with your smartphone. Smartphones are slowly becoming the best human companion.

Smartphones bring some new tools and methods to do things along with new technologies. They offered new opportunities in different fields such as the sales area with e-commerce and online shopping (M., 2017). Therefore, there is no reason that teaching doesn't take advantage of this new technology and all the opportunities it has to offer.

Students spend more and more time on their smartphones and less time studying. In this paper, we thought about a new and different way to provide learning content to students. Instead of asking them to come to classes and give them paper materials, we want to offer them the possibility to learn and study at their own pace at their preferred moment.

To do that, we want to supply them courses materials directly on their smartphone. By doing that, we ensure that students will be able to study when they want and what they want. We release them from the original constraints of being present at a specific day in the week at a specific time. They are now able to study 24/7, as much as they want and as often as they want.

Every student is different and has their own preference regarding the time they want and need to spend on some theory slides and exercises as well as when and where they want to do it. By providing them course content on their mobile phone, we offer them the possibility and freedom to study whenever they want.

In addition to that, we are now able to retrieve every piece of information regarding a student and his learning profile. We can retrieve data regarding his progression, performance and interest for each course he is involved in. That is something we simply cannot do using the traditional method where we teach simultaneously 30 students at the same time.

This new way of teaching and learning is both profitable for students, who find themselves freer in their studies and for teachers, who can track every student information and can adapt their teaching method and course content.

Data is extremely important and precious. As we retrieve it, we do not simply can use it to track student progress and adapt course content but we can also use it with artificial intelligence to provide each student personalized study recommendations.

Students who are engaged in a degree's program have to choose courses every semester for the next one. It is not always easy as they often don't know exactly what they want and they don't know exactly what best suits them. We often see students choosing courses that they finally drop because they realize that they don't like it and that is not what they want to do. It is perfectly normal and it is part of the life but it would be much more interesting if they could immediately find what they like and what they want to do.

Every degree's program is composed of different paths that contain a lot of courses. Each course is unique but courses that belong to the same path are often pretty similar.

The goal of this thesis is to determine which path is the most and the best suited for each student so we can ensure that they choose courses they will like and that meet their expectations in order to avoid a potential reorientation.

Based on the data we collect and with the help of artificial intelligence and machine learning, we can recommend the study path and courses to students.

In addition to that, we provide teachers a new and different way to give lectures. Via the website we developed, teachers can create lectures, add external links as resources and create quizzes. They can gather these resources per week that they can also create, modify and delete. For each course, teachers can have a quick and detailed overview of participants' results, progression and commitment. Based on that, they can quickly and easily contact students and/or adapt their course.

## **2 Research questions and methodology**

### **2.1 Questions**

This thesis tries to answer multiple questions as we don't only focus on the mobile learning and teaching methods but also on the data processing and the use of artificial intelligence.

However, the following questions are directly related to this study:

- How to provide a study path recommendation for students with the help of artificial intelligence?
- How to transform a student learning activities and experience into indicators for teachers?
- How can teachers adapt their teaching methods based on students' learning outcomes?
- How can a mobile application help and motivate students learning and studying?

In order to answer these questions and try to find solutions and opportunities, we developed a mobile application for students where they can access course content such as PDF lectures, videos, website links and quizzes. We decided to focus on the Business Information Technology Degree's program dispensed at Haaga-Helia.

We created three different modal courses from three different paths. Firstly, we offered an introduction course on the JavaScript programming language. This course belongs to the "programming" path. Secondly, we proposed a course on UX Design that typically belongs to the "design" path. Finally, we created a course on Project Management that focus on Agile & SCRUM methodologies. This management-oriented course belongs to the "business" path.

Each course spans 5 weeks. Every week, there is content theory, videos, additional lectures and quizzes.

### **2.2 Methodology**

#### **2.2.1 Test case**

In order to test the application's algorithms and to prove that its recommendation system was correctly working, we also run a series of tests with sample students. The last test we

ran was a test case where we created 25 different students and inserted them in the data-base with random values as data.

We simulated a typical class composed of different students with different skills and abilities. We wanted to verify that the recommendation the application would make was correct and plausible. We used these students as a sample of our target population. We needed to prove our previous results and that the application was working as we expected.

	A	B	C	D	F	G	H	I	J	K
	Firstname	Lastname	Student number	Email	Course	Time spent (ms)	% Done	Global Score	Preference	Recommendation
2	Lydia	Pace	a1572960	lydia.pace@email.com	JavaScript	5000	90	90	1	
3					UX Design	3500	80	70	2	
4					Agile project management	1000	20	10	3	
5	Salma	Erickson	a1572961	salma.erickson@email.com	JavaScript	6000	80	80	1	
6					UX Design	4000	80	80	2	
7					Agile project management	2000	80	80	3	
8	Nickolas	Moran	a1572962	nickolas.moran@email.com	JavaScript	5000	90	90	1	
9					UX Design	7000	70	70	2	
10					Agile project management	10000	10	10	3	
11	Madeline	Mendez	a1572963	madeline.mendez@email.com	JavaScript	10000	95	95	1	
12					UX Design	5000	100	50	2	
13					Agile project management	2000	20	30	3	
14	Giselle	Harrison	a1572964	giselle.harrison@email.com	JavaScript	1000	20	30	3	
15					UX Design	2000	40	60	2	
16					Agile project management	3000	90	90	1	
17	Raegan	Miles	a1572965	raegan.miles@email.com	JavaScript	5000	50	50	3	
18					UX Design	6000	60	60	2	
19					Agile project management	7000	70	70	1	
20	Mia	Villegas	a1572966	mia.villegas@email.com	JavaScript	10000	50	80	3	
21					UX Design	9000	90	80	2	
22					Agile project management	8000	95	85	1	
23	Adam	Branch	a1572967	adam.branch@email.com	JavaScript	5000	80	80	3	
24					UX Design	5000	82	80	1	
25					Agile project management	5000	81	80	2	
26	Luca	Alexander	a1572968	luca.alexander@email.com	JavaScript	6000	80	80	3	
27					UX Design	6000	80	81	2	
28					Agile project management	6000	80	82	1	

Figure 1 : Sample students data

D	F	G	H	I	J
Email	Course	Time spent (ms)	% Done	Global Score	Preference
lydia.pace@email.com	JavaScript	5000	90	90	1
	UX Design	3500	80	70	2
	Agile project management	1000	20	10	3

Figure 2 : Student 1 data

```

1  _id: ObjectId("5cd18da6cf213d001a4f216d")
2  firstName : "Lydia "
3  lastName  : "Pace "
4  number    : "a1572961 "
5  email     : "lydia.pace@email.com "
6  password  : "sha1$77a9f656$1$e3952fa969cf8d6198473eaf670632ef0a9cd24b
7  courses   : Array
8    0 : Object
9      _id: ObjectId("5cd18da9cf213d001a4f2171")
10     courseId : "5c8a1ba0f78a770019265865 "
11     path : "Programming "
12     globalScore : 90
13     > globalResults : Array
14     > quizResults : Array
15     timeSpent : 5000
16     > done : Array
17     percentage : 90
18   1 : Object
19     _id: ObjectId("5cd18da9cf213d001a4f2170")
20     courseId : "5c8b7036acfabb001aa351d3 "
21     path : "Design "
22     globalScore : 70
23     > globalResults : Array
24     > quizResults : Array
25     timeSpent : 3500
26     > done : Array
27     percentage : 80
28   2 : Object
29     _id: ObjectId("5cd18da9cf213d001a4f216f")
30     courseId : "5c8f93e537b8a8001a60ec6e "
31     path : "Business "
32     globalScore : 10
33     > globalResults : Array
34     > quizResults : Array
35     timeSpent : 1000
36     > done : Array
37     percentage : 20
38   __v : 0

```

*Document Modified.*

Figure 3 : Student 1 in the database

As we can see in the figures above, we generated different values for each course and for every student and ranked their favourite courses by these values. We left the recommendation column empty and logged in the application after inserting the data in the database in order to check what the system recommended to that specific student.

Next, we wrote down the course rank in the recommendation table in the application. We were then able to compare the student real preference and most optimized recommendation based on a human interpretation of these values and what the system actually recommended. The goal was to prove that the system, without any help, would recommend the

same courses in the same order as what a human would recommend. It would be the proof that the algorithm is correct and the artificial intelligence has been rightly implemented and works correctly.

The results of this test case will be discussed later on this paper in the results section.

### **2.2.2 Study case**

To conduct this research, we offered this application to 5 users who were all students of the Haaga-Helia University of Applied Sciences. Their mission was to test the usability and user experience of the mobile application. Students were in the first or third semester of their studies.

Firstly, users had to answer 3 pretest questions such as “Have you already used your phone for educational purposes ?” or “From 1 to 5, how confident are you using new applications ?”.

Then, students had to do nine different tasks to test the usability. Some test cases were given to them. Test cases included: log in the application, search a course, open any PDF for that course, take a quiz, check progress, etc.

Finally, users had to answer different questions on a post-test interview regarding the usability and user experience. Some questions included: “What features did you think were the most important and useful for you ?” or “Was there something you found difficult or frustrating ?” but also “What was your general impression of the app ?”, etc. In addition to that, users had to grade from 1 to 5 how accurate the app was to some adjectives such as fun, interesting, boring or complicated. In this post-test interview, students also gave their suggestions and recommendations. They also graded how likely they would use the application if it was currently available.

The goal of this post-test interview was to determine whether the application would be a helpful addition to the students’ daily life. The tests were supervised by a note taker that had for mission to track time students spent on the different tasks and take note about the users’ body language. The only sign they received was to test the application using the given credentials and to take notes if certain tasks would happen to fail.

The results of this research will be discussed later on in this paper in the results section.



### 3 Related research

This thesis is based on the paper “*Towards an Adaptive Study Management Platform: Freedom Through Personalization*” (Dirin & Laine, 2018) from Dr. Amir Dirin and Dr. Teemu. H. Laine. The paper from Dr. Dirin and Laine tries to propose a common and adaptive platform for both students and teachers to personalize study management and provide analysis services regarding students. Our research paper focuses on the second and third phases of the mentioned research which are when the student start studying at the university and when he has to select a study path.

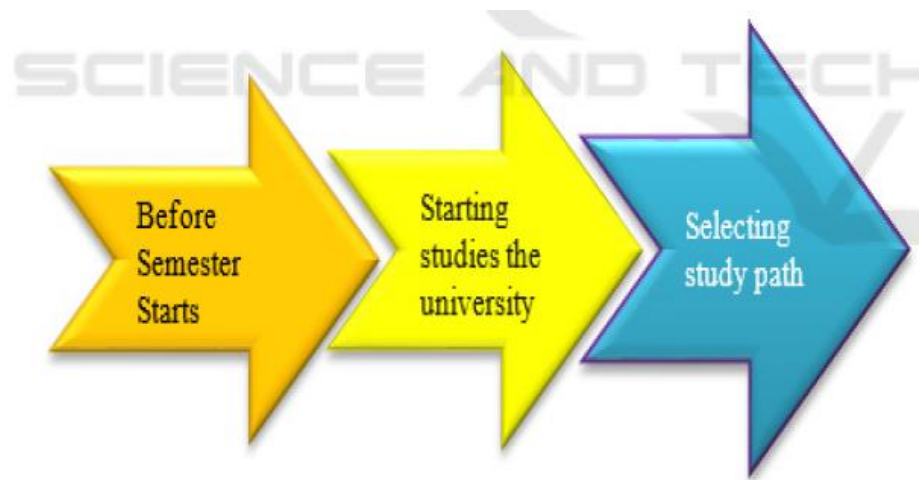


Figure 4 : The 3 different study phases based on Dr. Dirin and Laine's research (Dirin & Laine, 2018)

Some other papers mentioning the problematic of teaching and studying in a mobile context that are worth a read :

- ☞ “*Mobile learning: A framework and evaluation*” by Mr. L. Motiwalla
- ☞ “*Guidelines for learning/teaching/tutoring in a mobile environment*” by O’Malley, Vavoula, Glew, Taylor, Sharples, Lefrere, Lonsdale, Naismith and Waycott
- ☞ “*The Evolution of Mobile Teaching and Learning*” by Mr. Guy Retta
- ☞ “*Mobile Learning: Teaching and Learning with Mobile Phones and Podcasts*” by Mrs. A. Moura and Mrs. A. Carvalho
- ☞ “*The incorporation of mobile learning into mainstream education and training*” by Mr. D. Keegan

The first paper wrote by Mr. Motiwalla explores the learning behaviour of mobile learning and experiential learning. Mr. Motiwalla performed on-site observations on youths in a

College Exploration Camp in Taiwan. He wanted to evaluate the effectiveness of learning when combining mobile learning with experiential learning activities.

“The Evolution of Mobile Teaching and Learning” is a book written by Mr. Retta that encompasses three different sections. The first one includes different detailed definitions of mobile learning and provides theoretical background of distance education. The second section shows results of pilots, projects and trials relevant to the use of mobile teaching and learning. Finally, the last part assesses the future of mobile education. This paperback is a very detailed and complete work.

The research paper from Mr. Keegan provides definitions of mobile learning while presenting major projects and giving examples of incorporation of mobile learning into the mainstream. This research paper is a good introduction on mobile learning as it is a well-popularized approach on mobile learning in general.

Some papers addressing the question of how to recommend/suggest something with the help of the artificial intelligence or based on big data :

- ☞ *“Personalized Links Recommendation Based on Data Mining in Adaptive Educational Hypermedia Systems”* by C. Romero, S. Ventura, J. Delgado and P. De Bra
- ☞ *“A music recommendation system based on music data grouping and user interests”* by H. Chen and A. Chen
- ☞ *“Knowledge warehouse: an architectural integration of knowledge management, decision support, artificial intelligence and data warehousing”* by H. Nemat, D. Steiger, L. Iyer and R. Herschel
- ☞ *“Artificial Intelligence and Environmental Decision Support Systems”* by U. Cortés, M. Sanchez-Marrè, L. Ceccaroni, I. Roda and M. Poch.

The article “A music recommendation system based on music data grouping and user interests” is my favourite one from the list above as it is very similar to what we tried to achieve with this paper. The only difference lies in the fact that it involves music instead of learning. However, the recommendation system is pretty similar.

Mr. H. Chen and Mr. A. Chen designed a music recommendation system that provides personalized service of music recommendations. We did the same with a personalized service of courses recommendations. Mr. Chen analysed users’ access histories to derive user interests. They used content-based, collaborative and statistics-based recommendation methods. In addition, they carried out a series of experiments in order to prove that this approach is feasible.

It is an extremely interesting paper that is definitely worth a read. It helped me a lot when thinking about the design of the application and the recommendation system. Also, it motivated me to continue this research.

“Personalized Links Recommendation Based on Data Mining in Adaptive Educational Hypermedia Systems” is a paper describing a personalized recommender system that uses web mining techniques for recommending a student which links he should visit next. They present a specific mining tool and recommender engine that they integrated in the Aha! system in order to help teachers to carry out the whole web mining process. They reported on multiple experiments with real data to prove the suitability of using mining algorithms for discovering personalized recommendation links.

This paper is far more complex than the others and goes deeper in details in the mining algorithms. What is interesting, though, is that the authors integrated their recommender engine in a system used by teachers. The recommendations are intended for students like our project.

The scope of the related research to our paper can be quite large as it includes several questions and addresses several themes such as the **mobile teaching/learning** and the help in decision-making based on raw data with the help of **artificial intelligence**.

Except from the original research from Dr. Dirin and Laine on which this research is based on, there is no other research that treats all these themes at the same time. However, every research paper speaking about mobile learning or recommendation and suggestion based on data collected can be related in a way to this thesis and therefore is interesting for us.

## 4 Design

### 4.1 Mobile Application

Regarding the design of the mobile application, we tried to stay clear, sober and straightforward. Every big theme such as the course overview or info or achievements is addressed its own specific screen. Each screen includes required details but no unnecessary information. The goal of the application is to stay straightforward and simple. For that, we used a sober and basic design.

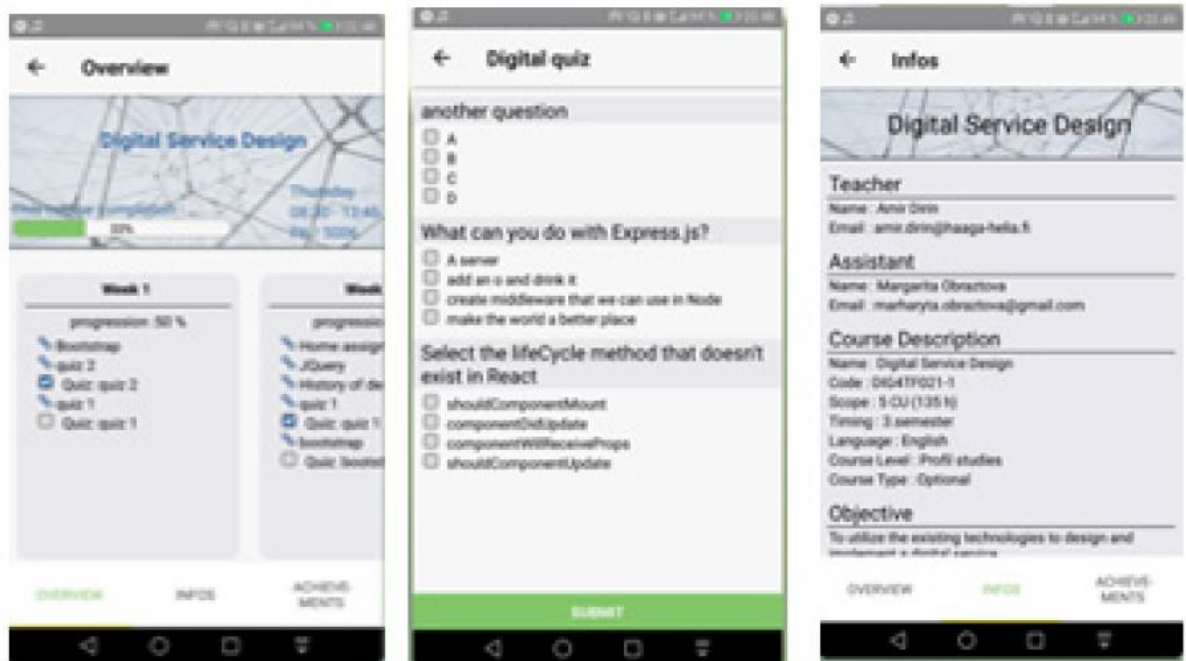


Figure 5 : Samples of phase 2 prototype screenshots designed by Dr. Dirin and Laine (Dirin & Laine, 2018)

The way the navigation works is similar to the existing navigation students already know with Moodle on their computer. There is a dashboard screen with all the students' courses listed and that can be accessed with a simple click. The idea was to give the student an overview of all his courses and to gather all of them at the same place.

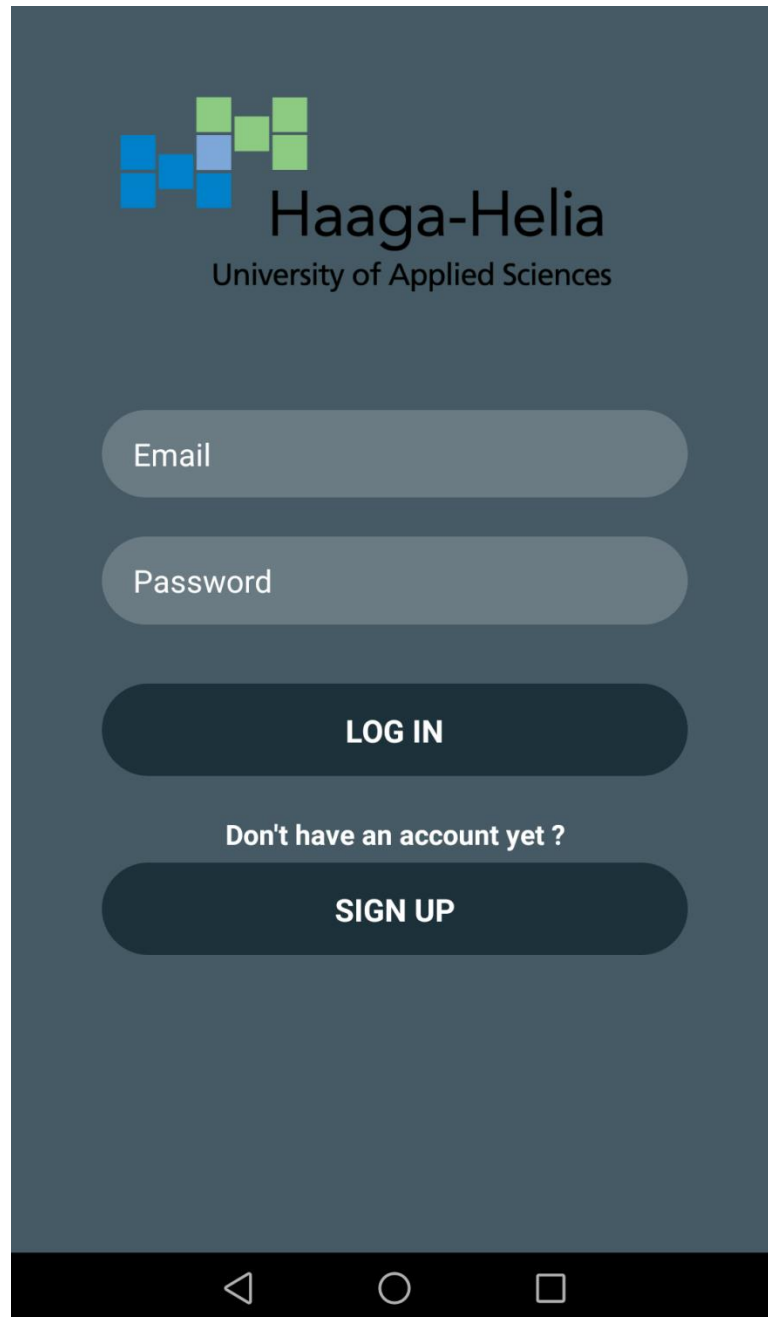


Figure 6 . Log in screen

The figure above shows the log in screen which is also the welcome screen. When you open the mobile application, you end up on this screen where students have the possibility to log in using their email and password as credentials.

If the student doesn't have an account yet, he has the possibility to sign up and create an account instead.

Haaga-Helia  
University of Applied Sciences

First name

Last name

Student number

Email

Password

Repeat Password

**SIGN UP**

Figure 7 : Sign up screen

The figure 4 shows what the sign up screen looks like. It is a simple form that the student has to fill in order to create an account.

The student has to write his first name, last name, student number, email address and choose a password. Once it's done, the student can click on the "sign up" button and will be redirected to the sign in screen where he will be able to log in.

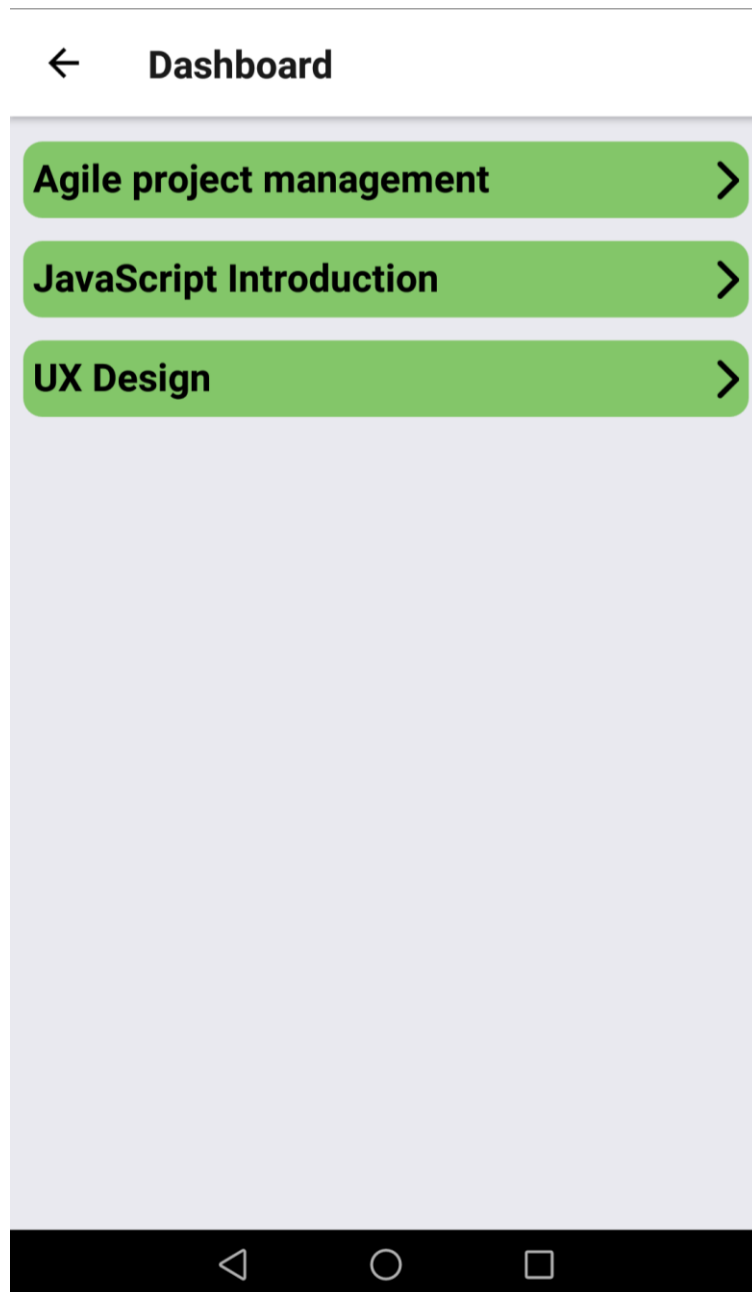


Figure 8 : Dashboard screen

Each course has its own screen. The content, details, information, achievements of the course can be accessed via the bottom tabs. That is a quite easy and practical way to split the information and a quick and easy way to navigate. At the top of every screen, there is a title that indicates where the user is (which tab, which course and on what he clicked on).

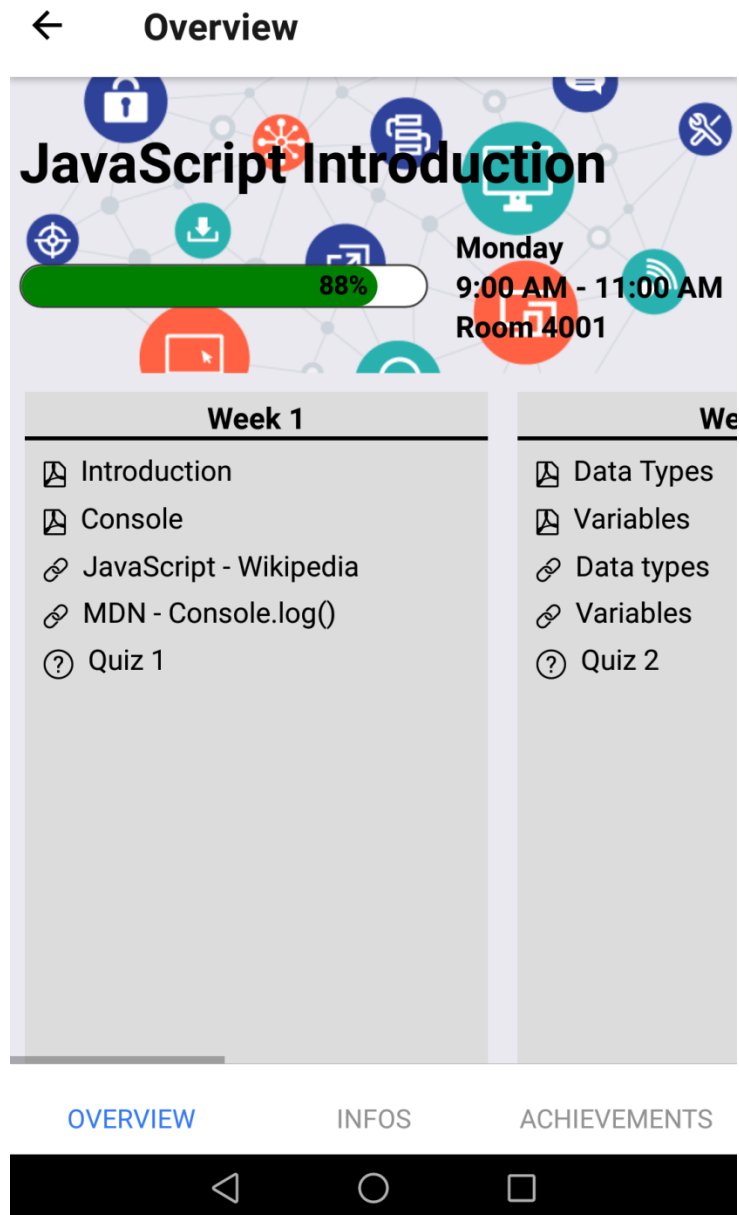


Figure 9 : Example of a course screen (JavaScript Introduction)

The overview tab of the course is split in 2 parts. In the top part that we can call the header, we can find basic information about the course itself such as the name of the course and its schedule including the day and the hour the course takes place. At the bottom left of the header, a progress bar indicates the students its progress for that course only by giving him the percentage of what he has accomplished.

The bottom part, that we can call the body part, contains the course content that is separated in weeks. Here again, we tried to keep the same structure as what students can currently find on Moodle. The course content is split into weeks and each week contains its lectures, links to external resources and quizzes. For the weeks we used a horizontal



scroll bar as it is more convenient, practical and user-friendly than a screen for each week.

The infos tab contains all information about the course itself. That includes the teacher's details such as its name and email in case the student need to contact him. It also contains the course description which includes the code, scope, timing, language, level and type of the course. The objective of the course is also described in this screen.

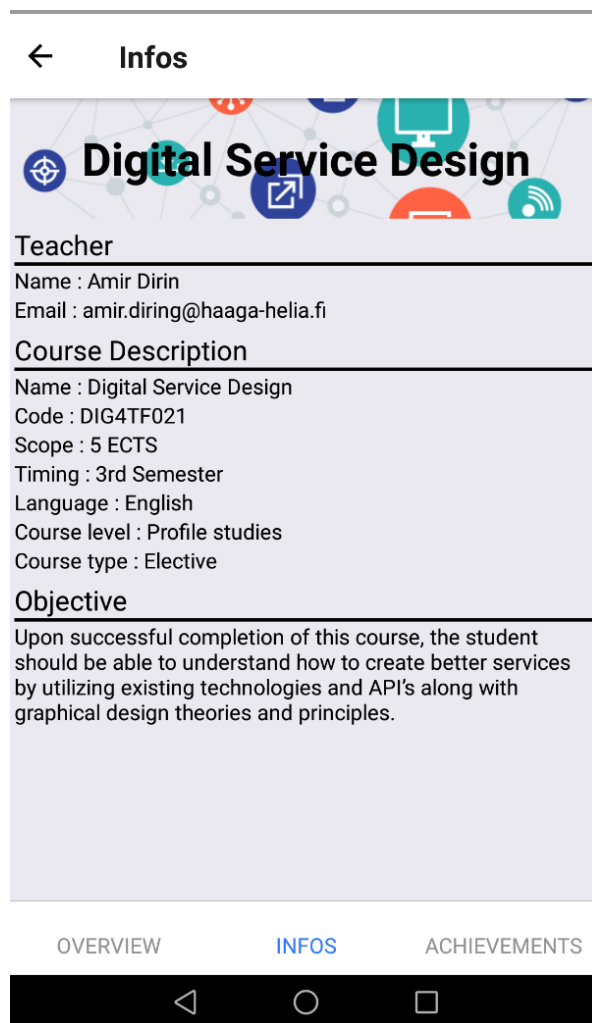


Figure 10 : The infos tab giving the course's information

The achievements tab gathers all quizzes' results. Here, the student can see what he has done and how good he has done. In addition to that, the student can discover the time he spent on that course reading lectures, taking quizzes, etc. An average percentage is calculated based on the results of every quiz he took.

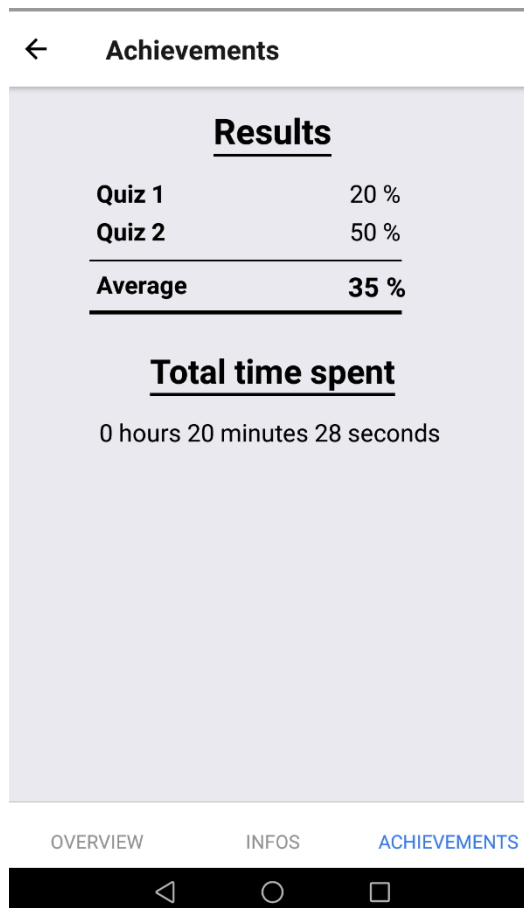


Figure 11 : The achievements screen with results' details

The quiz screen display all the questions and their possible answers. Questions are multiple choice questions. They all have four possible answers but only one is correct. The number of possible answers or questions is unlimited and can be dynamically adjusted. At the bottom of the screen, there is a fixed submit button that is used to submit the quiz answers when the student has filled it.

Note that the questions and their answers are mixed every time and therefore for each student. No student will have the exact same quiz as his colleague. The question 1 could be question 3 for another student and the order in which the answers are displayed is always different. This is done on the client side of the application.

Apart from that, the student can only take the quiz once. The number could be changed based on the needs but when the user opens the quiz, this information is sent to the server and saved in the database meaning that the student cannot retake the quiz as many times as he wants to modify his result.

← JavaScript Introduction - Quiz 1

What is JavaScript ?

- A programming language
- A request language
- An extensible markup language
- A markup language

Select the correct syntax to print the text "Hello" in the console.

- consolelog("Hello")
- console.log("Hello");
- console.log["Hello"];
- console.log(Hello);

In the browser, what is the keyboard shortcut to view the console ?

- F9
- F10
- F11

SUBMIT

Figure 12 : Quiz screen with multiple-choice questions

After taking the quiz, the student will not be asked to answer the questions again but will see his answers and his results the next time he clicks on the quiz link.

## ← JavaScript Introduction - Quiz 1

Results : 2 of 3

Percentage : 66.67 %

What is JavaScript ?

- A programming language
- A request language
- An exstensible markup language
- A markup language

Select the correct syntax to print the text "Hello" in the console.

- consolelog("Hello")
- console.log("Hello");
- console.log["Hello"];
- console.log(Hello);

In the browser, what is the keyboard shortcut to view the console ?

- F9
- F10
- F11
- F12



Figure 13 : A completed quiz with correct answers

Each lecture or external link opens in a different screen but in the same application.

Therefore, the user doesn't exit the application and information regarding the time he spent reading the material can still be recorded and saved. Also it eases the navigation and makes the user experience friendlier.

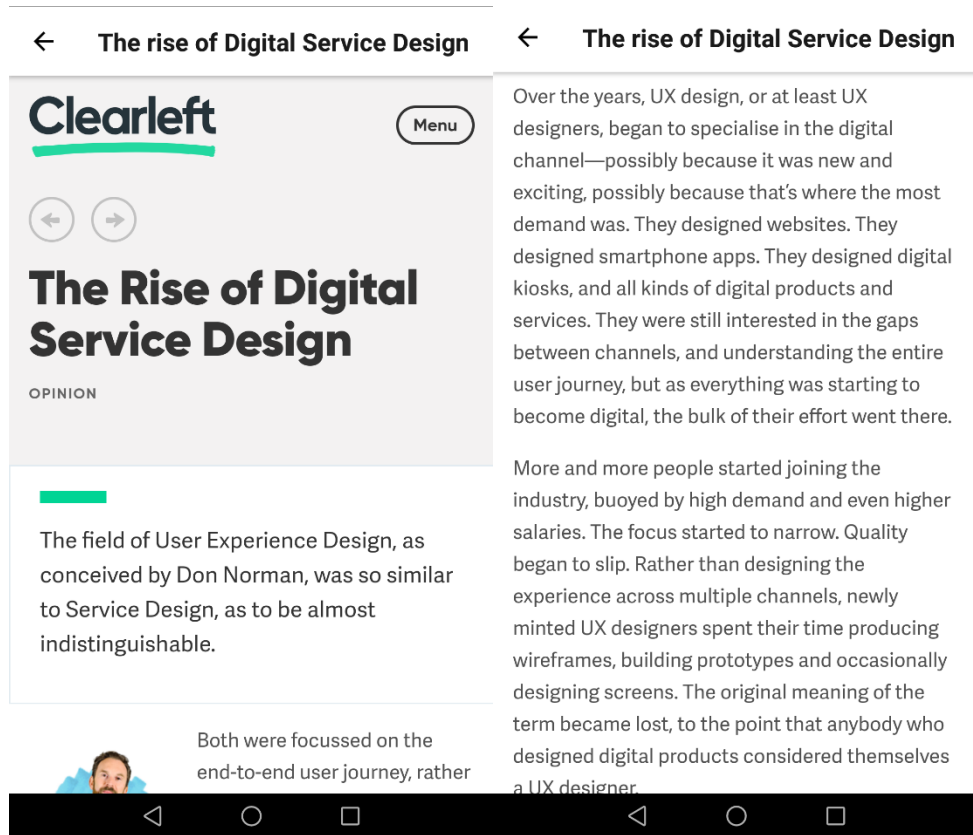


Figure 14 : External website displayed in the current application

The home screen simply displays the information about the student such as his name, student number and email address. This information could be useful for the student. For example, the student can find here his student number and cannot be edited on the application. The only way to modify it is on the website platform and requires an administrative role. The student could eventually change his profile picture directly from the application.

The home screen, in opposition to every other screen, has no “return” or “go back” button at the top left corner since it is the welcome screen. However, two buttons are displayed in the bottom part and allow the student to directly navigate to the dashboard screen and seeing all his courses or going to the recommendation screen.

## Home

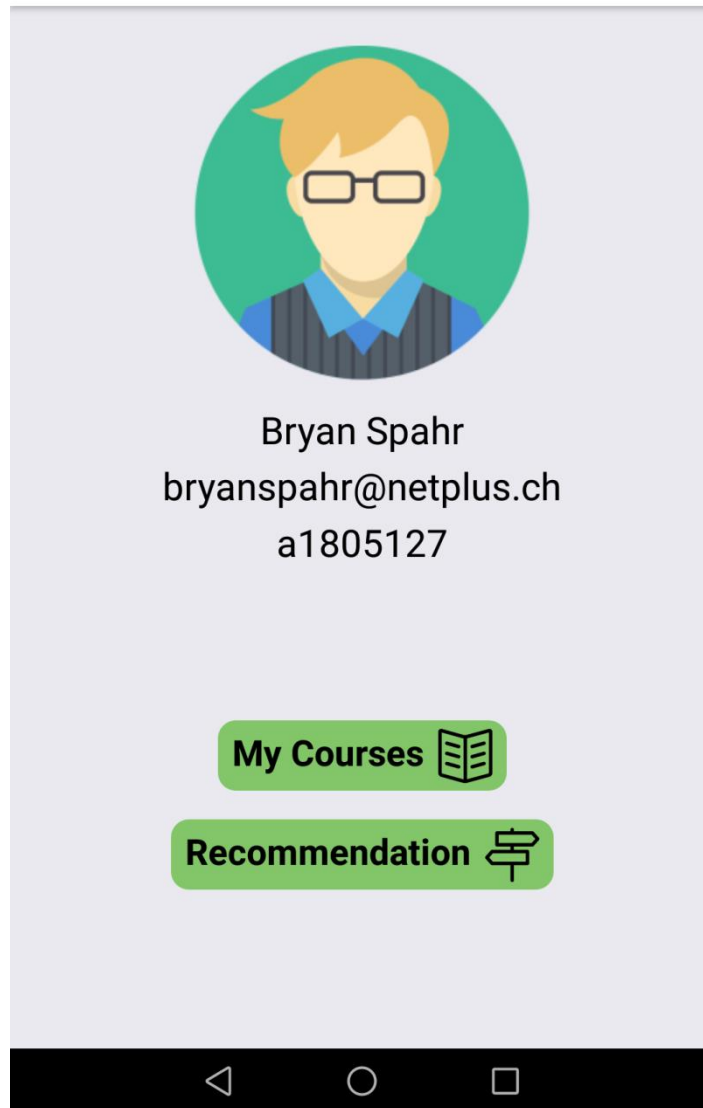


Figure 15 : Screenshot of the home screen

The recommendation screen is where the student can see what courses and path the application suggests him to choose but not only.

Here, all courses of the student are ranked based on their score. The score of each course is calculated with a formula that takes into account the average score of quizzes, the time spent reading the courses' materials and the percentage of the course content that has been opened and read by the student.

The ranking gives a clear and complete overview for the student of his performance and progress for each course as well as the time he spent on them. The total score is also transformed to a "grade" which is not the final grade for the course but a simpler and clearer way for the student to see how well he performed in a specific course. The higher

the score of a course is the better the grade is. The grading system is the same as the one used at school. The grades go from 1 for the lowest to 5 for the highest.

← Recommendation

**Favorites courses**

No	Name	Score	Done	Time	Grade
1	Mobile Programming	90	77 %	00:03:03	4.48
2	Server Programming	80	92 %	00:03:32	4.33
3	Database Developer	80	62 %	00:04:54	4.03
4	Business Intelligence	50	85 %	00:00:39	3.36
5	Digital Service Design	35	30 %	00:20:13	2.35

**Recommended path : programming**

Figure 16 : Recommendation screen with a ranking of the courses

Finally, at the bottom of the screen, the student can see what path it should choose his courses on for the next semester.

Each degree's program has different paths. For the Business Information Technology program for example, there are 4 different paths. Each path contains lots of courses. The higher the course is ranked, the most points it gives to its path. Finally, the path with the most points is recommended as the most suitable path for the student.

In clear, that means that the path of the courses where the student is the most performant and the most implicated is the most susceptible to contains other courses that could interest the student and where he would perform the best as well. For example, if the student

tends to have better results and spends more time on programming courses such as Mobile Programming or Server Programming, the application will recommend the student to choose courses in the programming path for next semester instead of courses in the design path.

## 4.2 Website

The design of the website is quite simple. We tried to stay sober and as functional as possible. The website is typically used by teachers and administrators only.

Teachers can access their course. They can create, modify and delete week. They can create, modify and delete any type of content such as theory slides, external links, quizzes, etc. They also can see the list of all students as well as students enrolled in their course. It is also possible to list all teachers and courses at Haaga-Helia.

Admin can do the same thing as teachers except that they have different rights. They have more permission when it comes to courses, students and teachers. They can create, modify and delete courses, students and teachers.

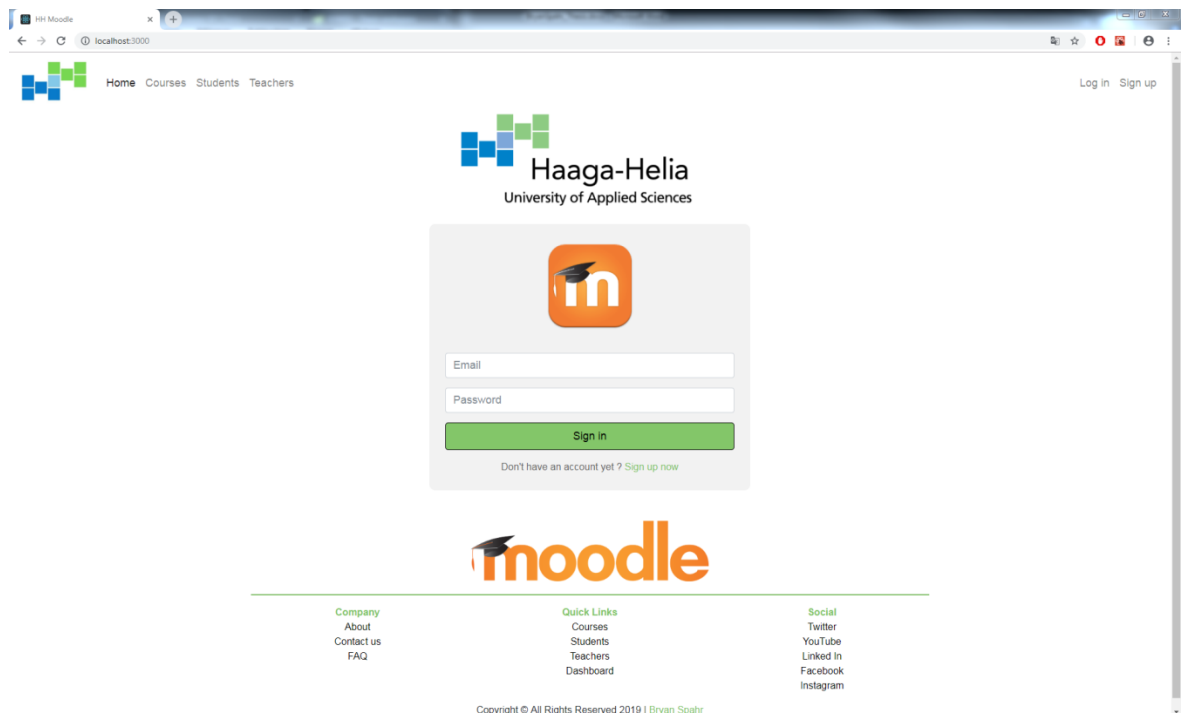


Figure 17 : Home page of the website

When landing on the home page of the website, you immediately have the possibility to sign in. Pages regarding the courses, students, teachers and other sensitive information are private and cannot be accessed unless you are logged in. If you try to access it anyway, you will be immediately redirected to the login page.



If the teacher doesn't have an account, he has the possibility to sign up and create an account like shown in the figure below.

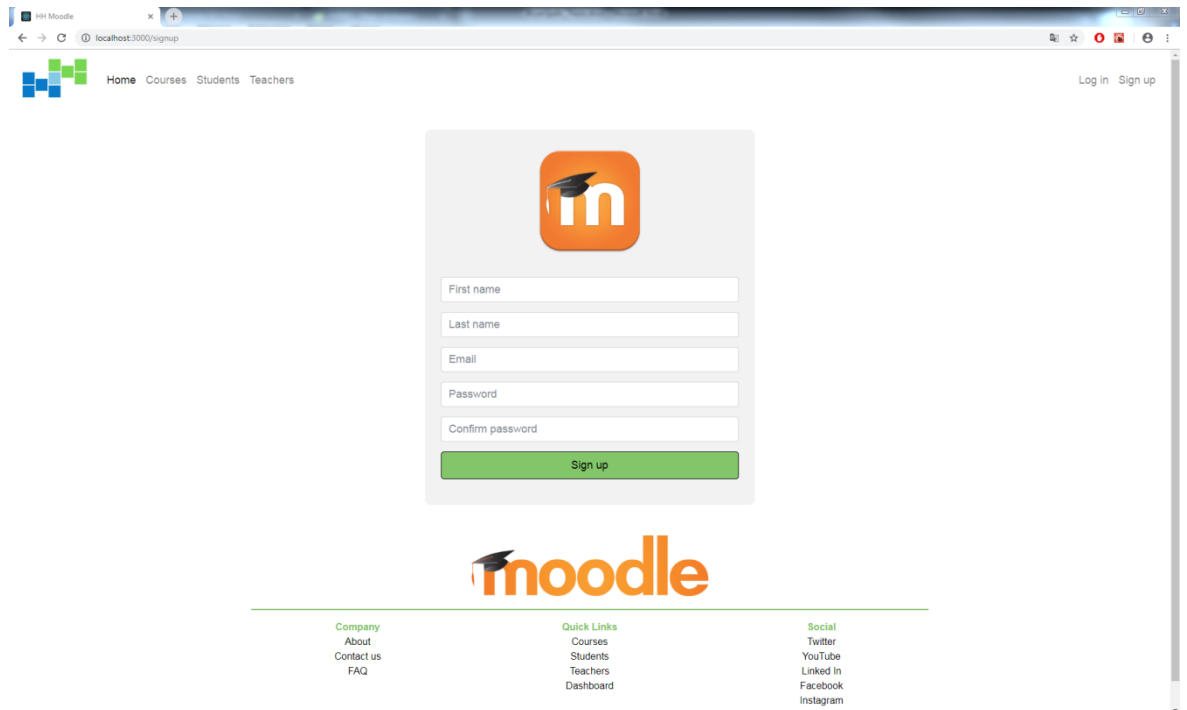


Figure 18 : Sign Up page

When connected, the teacher can have a look at all courses offered at Haaga-Helia. The courses are displayed in a table component with their name, code and language. It is also possible to search a course by its name or code thanks to the search bar at the top of the table.

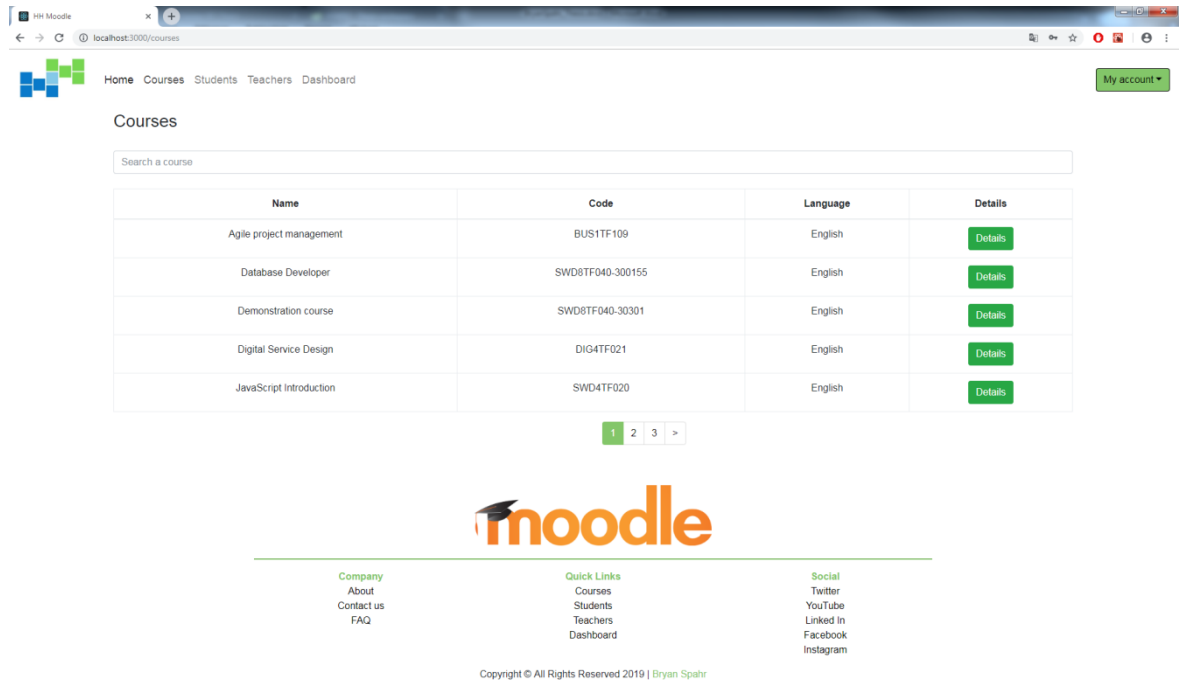


Figure 19 : List of all courses dispensed at Haaga-Helia

If the user connected is an administrator, he can see the same information as a teacher. The only difference is that he can quickly edit and delete a course. Also, with the admin role, it is possible to add a course with the help of a button on the bottom left corner of the table.

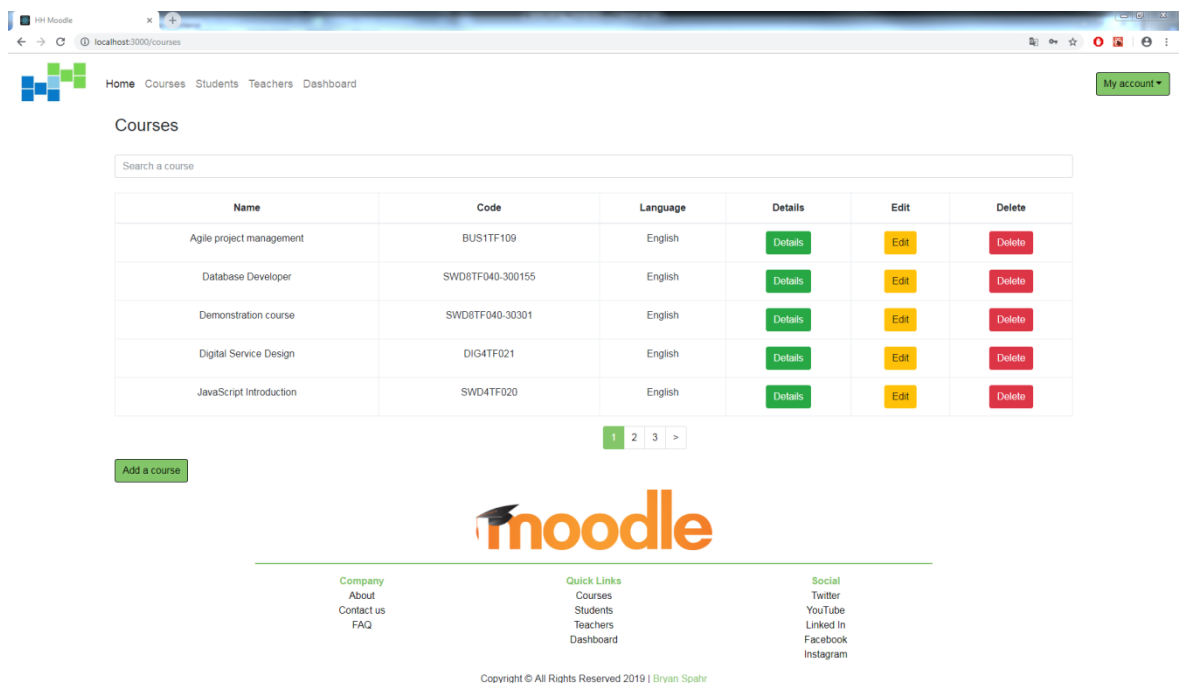


Figure 20 : Courses list from an administrator point of view

When clicking on the “details” button, it is possible to see more information about a course such as the scope, the timing, the path, the schedule and the objectives for example.

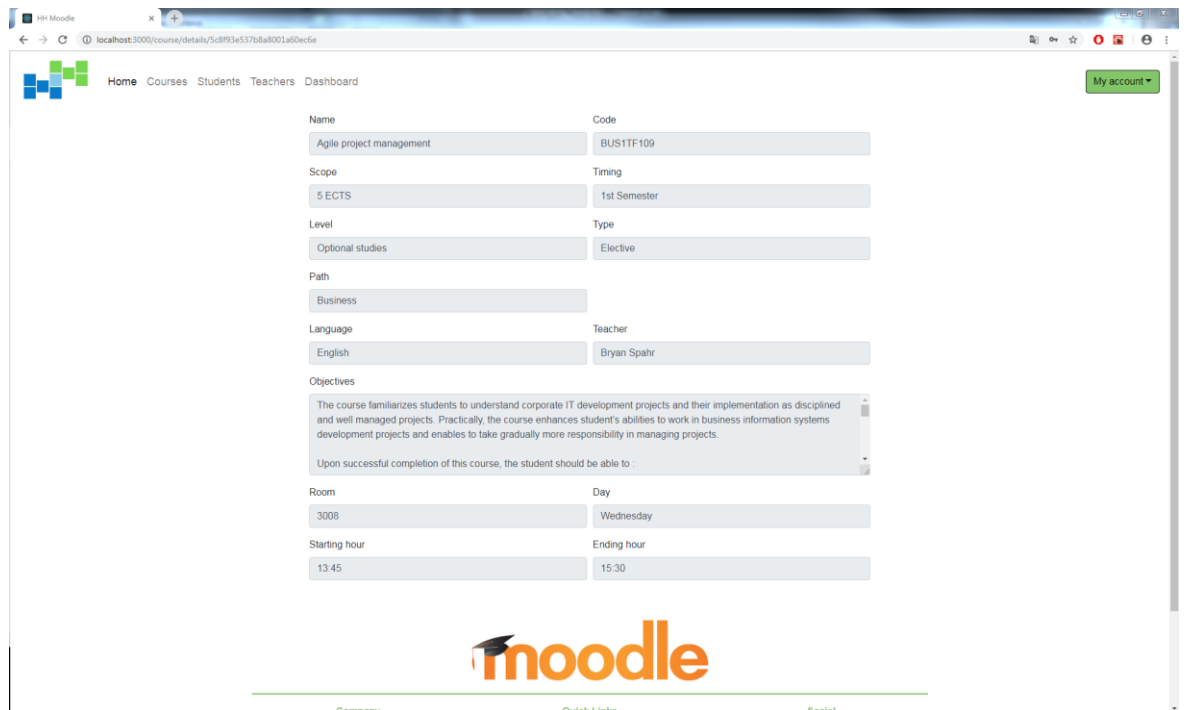


Figure 21 : Course details

This information can be edited and modified by an administrator. It is possible to write whatever we want in the objective text area or for the course code. However, the teacher must be selected from a drop-down list. This list is populated with the database meaning that a teacher must exist in the database before it can be chosen as the course teacher.

The form to add a course is similar to the form for editing a course except that all information are blank and have to be completed.

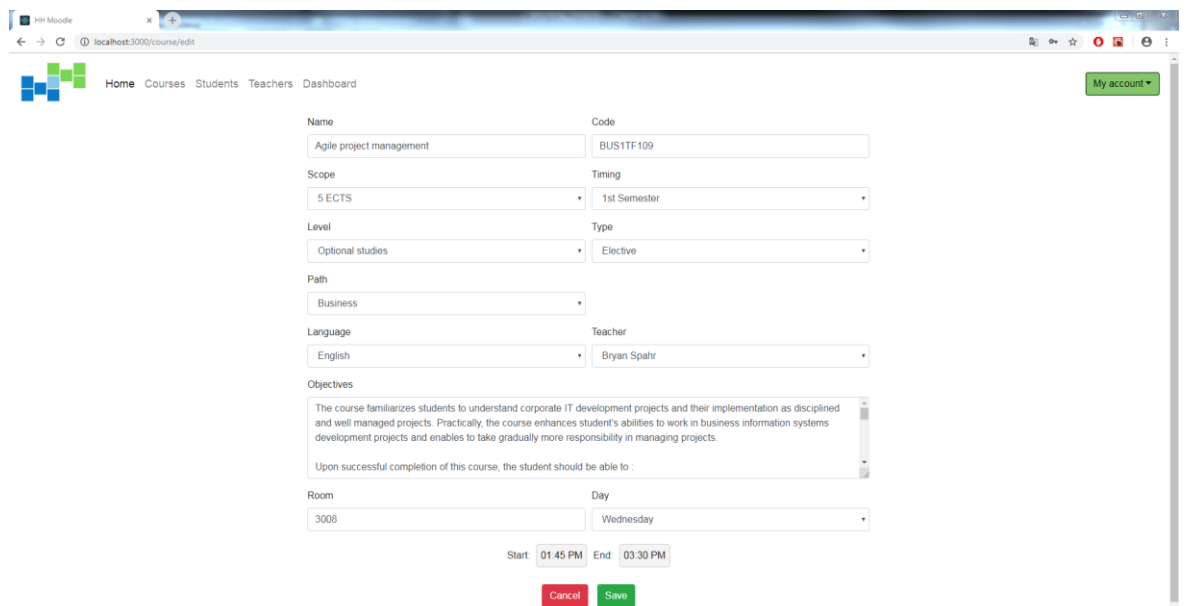


Figure 22 : Form to edit a course

The students tab is similar as the courses tab. There, the user can display the list of all students of Haaga-Helia in a table component that will show their first name, last name, student number and email address.

The functionalities related to the students are pretty similar to those related to the courses. It is possible to see them in detail and to edit or delete them for an administrator. Also, it is possible to add a new student or search for an existing one with the search bar component.

When adding a student, it is asked to give the student first name, last name, number and email address at least.

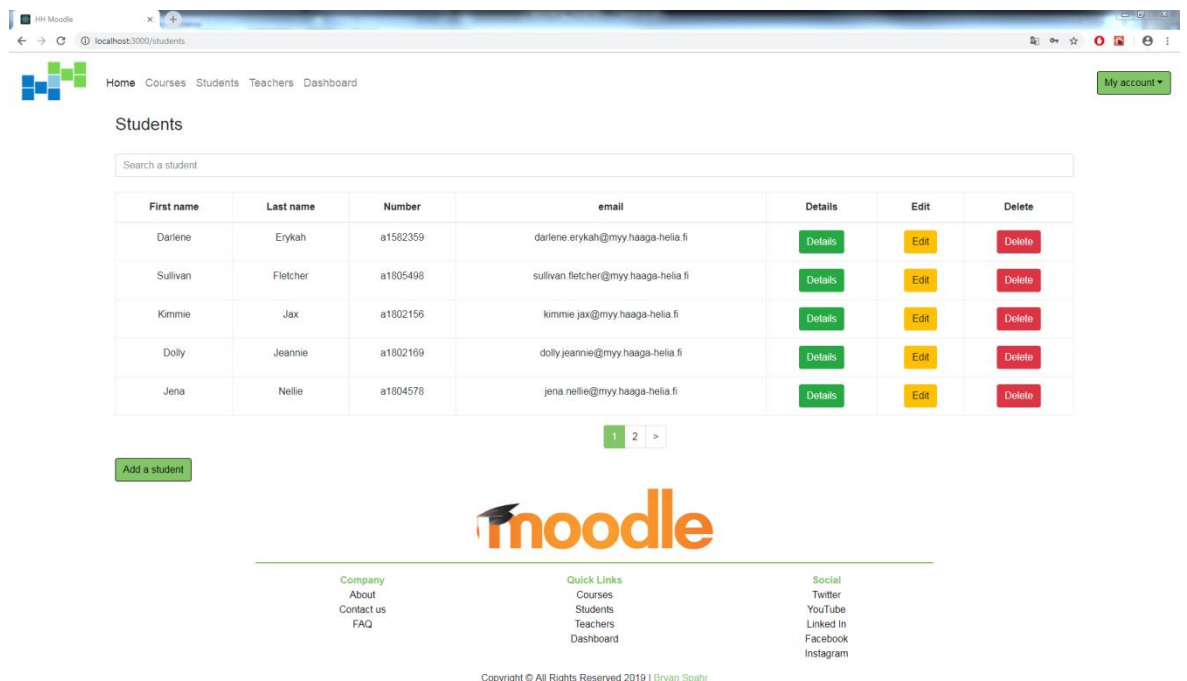


Figure 23 : List of students studying at Haaga-Helia

Another tab is the teachers tab. It is also pretty similar to the courses and students tab. The functionalities are the same and it is also the duty of administrators to add, modify or delete a teacher.

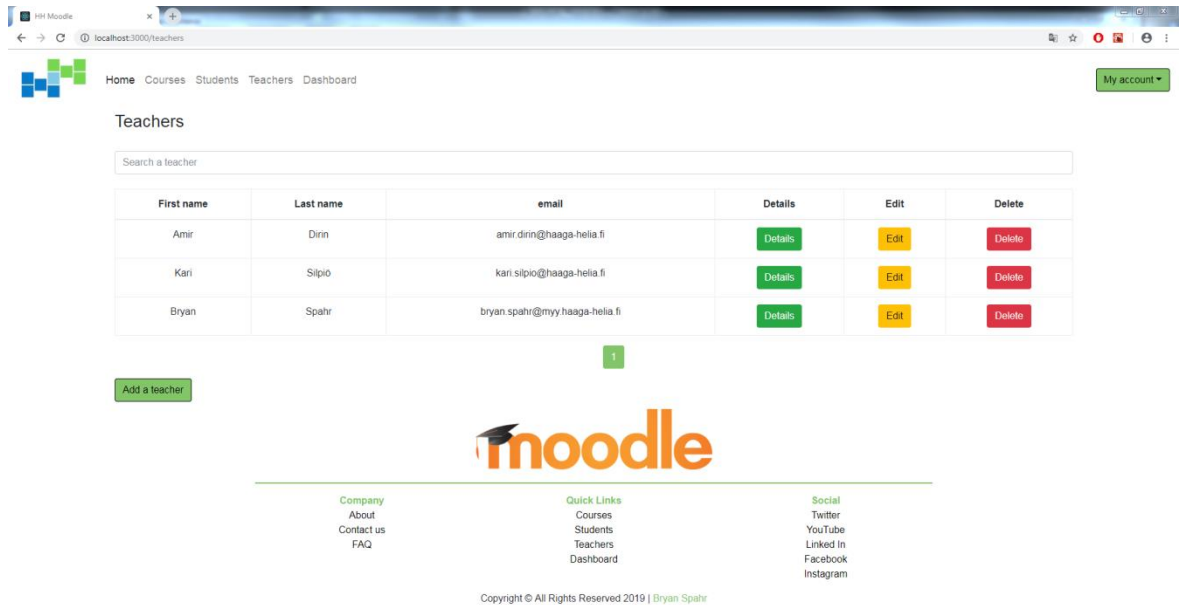


Figure 24 : List of teachers teaching at Haaga-Helia

The dashboard tab is a tab for teachers only. On this page, teachers can see and access directly all the courses they dispense. They don't see other courses and other teachers cannot modify these courses. Each teacher has its own personalized page.

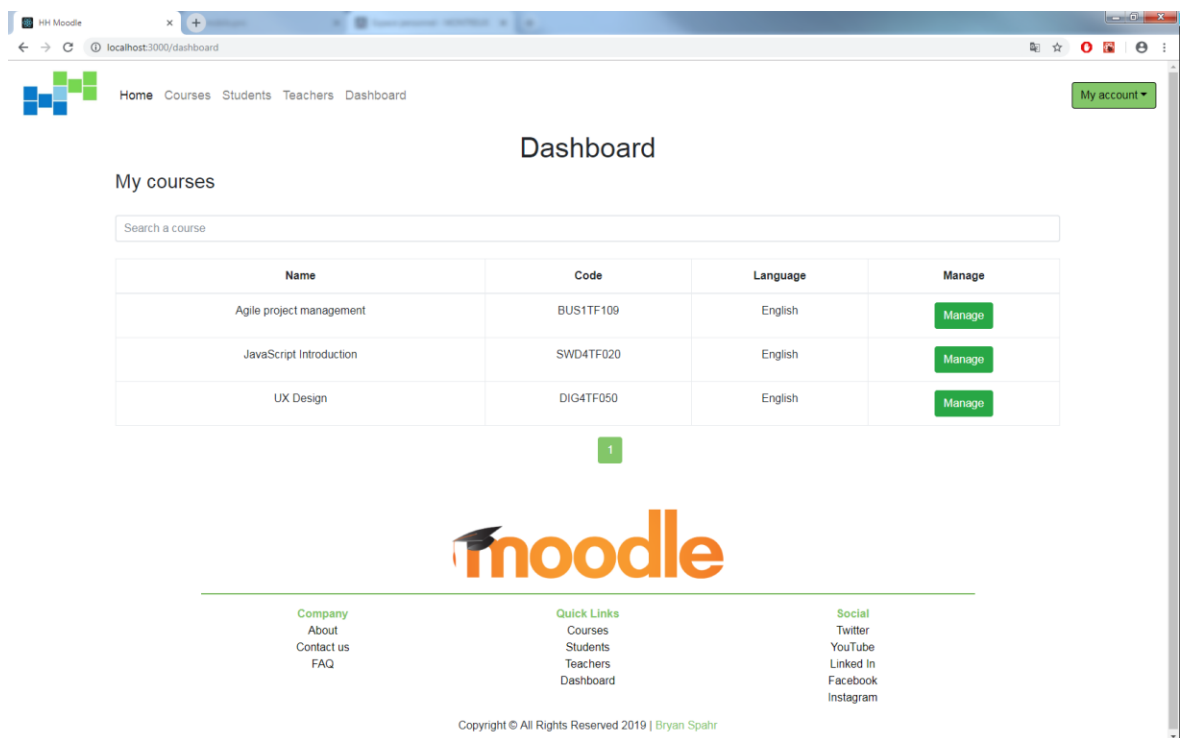


Figure 25 : Dashboard screen from the teacher point of view

When the teacher clicks on a course, he can access it and manage its content. Every course has the same structure. It is divided in weeks that the teacher can add or remove.

For every week, the teacher has the possibility to add a lecture, a link or a quiz. These resources are then created, saved in the database and gathered under the corresponding week.

This structure is pretty similar to what already exists now in the Haaga-Helia moodle website. It also the same structure that is used in the student mobile application. Therefore, it is easier for teachers to add, remove or modify specific content.

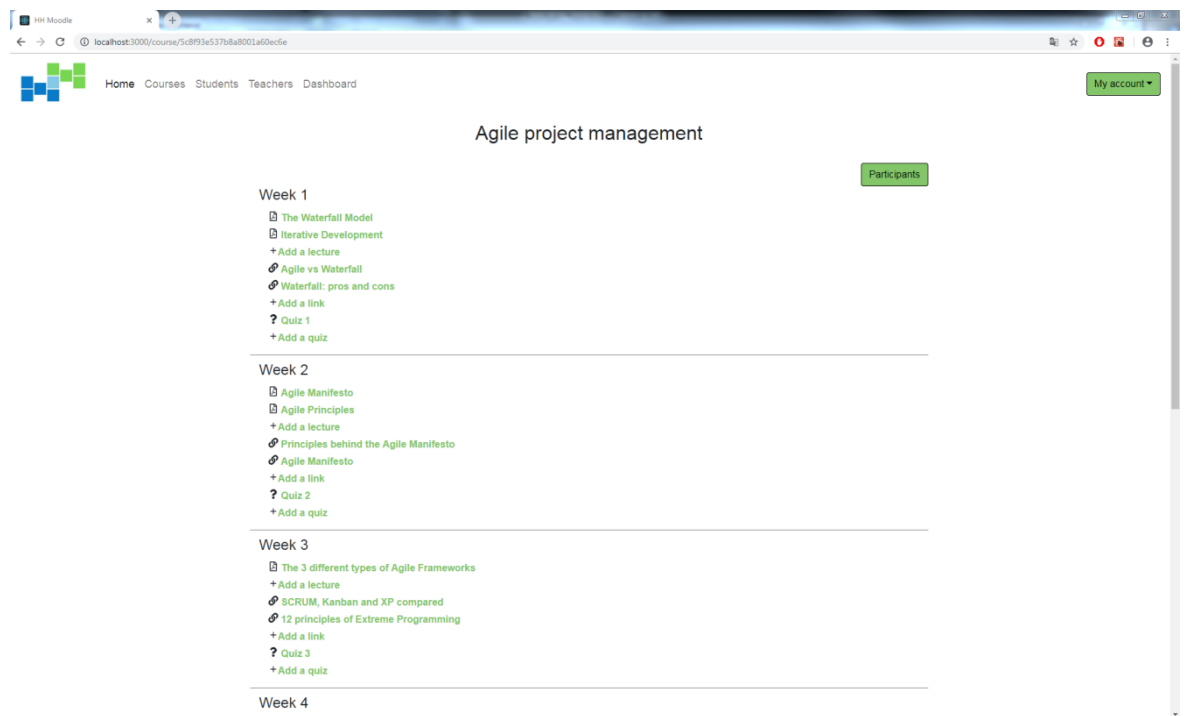


Figure 26 : Structure and content of the course organized in weeks

When creating a quiz, teachers should first give the quiz a name and indicate the number of questions they want.



Figure 27 : Creating a quiz

The next step to create a quiz is to write the quiz question and answers. There should be only one good answer and three wrong answers. The correct answer is always the first answer that needs to be written down but the answers are then mixed and displayed in a different order each time the quiz is open on the mobile application. Therefore, it is impossible to predict which answer is the correct one by its position without knowing it.

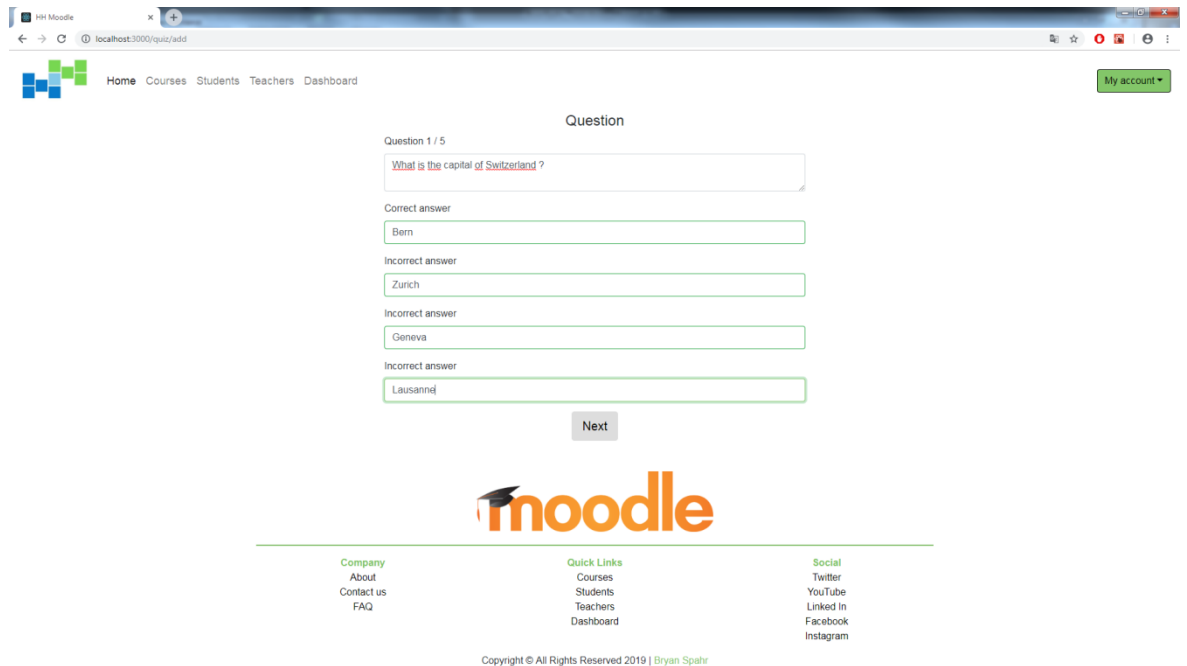


Figure 28 : Writing the questions and answers of the quiz

The process to add a lecture, video, image or external link is similar to adding a quiz. First, the teacher should choose a week, then he can add a resource by indicating the resource name (name of the website, title of the presentation, etc) and the resource link (the URL where the resource is hosted).

By adding a resource, its data are automatically saved and stored in the database. The resource will be immediately and dynamically available on the student's mobile application.

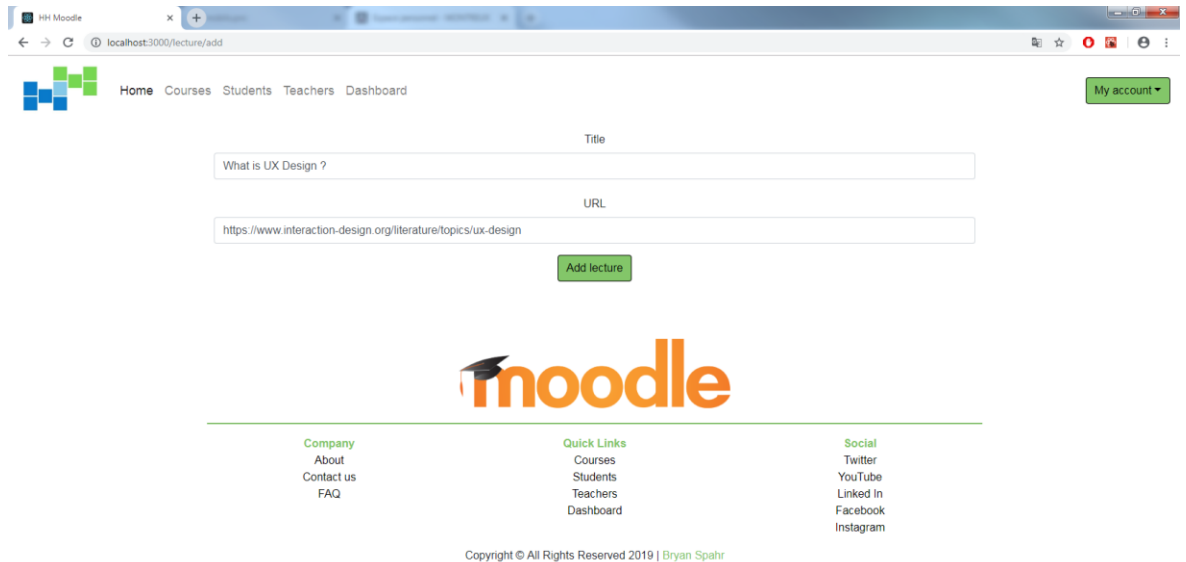


Figure 29 : Adding an external link as a resource

When the teacher created and structured his course, he can follow and track his students by clicking on the “Participants” button in the top right corner on the course page. This will display a table listing all students taking part in that course. With this table, the teacher can easily see the students’ main information and performance in that course. The table displays the first name and last name of the student, his progression in the course (the percentage of resources opened), his average score to quizzes and the time he spent on the course.

What is interesting here is that the teacher can easily filter the student, search a specific student or even sort the table by the percentage done, the average score (lowest or highest), etc. With that information, the teacher can have a quick and complete overview of his students.



The screenshot shows a Moodle course page for 'UX Design'. At the top, there are navigation links: Home, Courses, Students, Teachers, Dashboard, and a 'My account' button. Below this is a search bar for students. The main content is a table of participants with the following data:

First name	Last name	% Done	Avg. score	Time spent	Details
Dolly	Jeannie	76	60	00:01:29	<a href="#">Details</a>
Alexis	Stacey	41	41.66666666666666	00:00:47	<a href="#">Details</a>
Sullivan	Fletcher	52	19.999999999999996	00:00:52	<a href="#">Details</a>
Jena	Nellie	31	13.333333333333332	00:00:40	<a href="#">Details</a>
Bryan	Spahr	100	13.333333333333332	00:05:23	<a href="#">Details</a>

Below the table is an 'Add a student' button and a pagination control showing page 1 of 10. Underneath is a 'Leaderboard' section with the following data:

<b>Best student:</b>	Dolly Jeannie	<b>Average score:</b>	60 %	<b>Lowest student:</b>	M S	<b>Average score:</b>	0 %
<b>Most advanced student:</b>	Test Teste	<b>% Done:</b>	103	<b>Less advanced student:</b>	Gui Ctlj	<b>% Done:</b>	0
<b>Most dedicated student:</b>	Bryan Spahr	<b>Time spent:</b>	00:02:54	<b>Less dedicated student:</b>	M S	<b>Time spent:</b>	00:02:54

Figure 30 : Participants of the UX Design course

Also, under the participants table, the teacher can find a leader board showing the best student (the one with the best average score), the most advanced student (the one with the highest percentage of resources opened) and the most dedicated student (the one who spent the most time studying the resources). In opposition, he can also see the lowest student (the one with the lowest average score), the less advanced student (the one who opened the less number of resources from the available ones) and the less dedicated student (the one that spent the less time on the course).

The leaderboard allows the teacher to see the best and the worst students in a specific course in each category (studiousness, involvement, results).

The teacher has the possibility to click on any name displayed in the leaderboard or to click on the "Details" button on the last column of each row in the table. By doing that, the teacher is immediately redirected to the student profile and his study profile for that specific course.

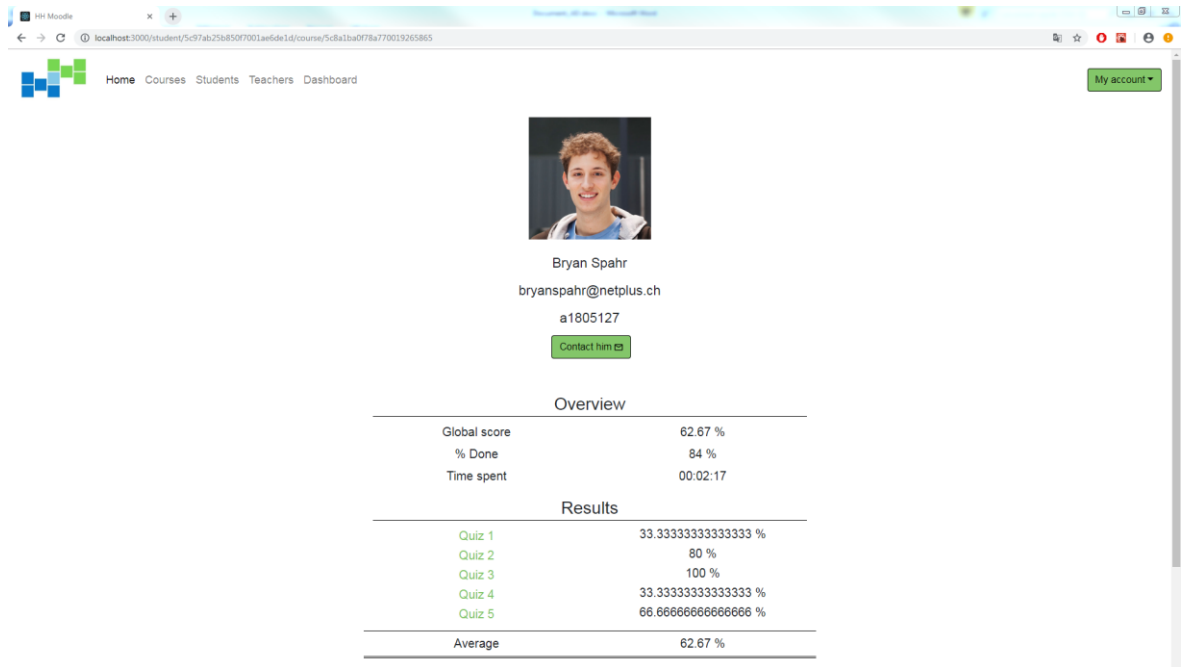


Figure 31 : Student course profile

In the figure above, the student profile of the student “Bryan Spahr” is displayed with some basic general information like his name, his email address and his student number. Below that, you can have a better a more complete view of the student profile regarding that specific course. You can see his global score, the percentage of resources he opened, the total time he spent on the course and more importantly his results for each quiz he took.

Here, you have a better understanding of the student results and progression in that course. It is also possible to quickly contact the student if needed by clicking on the “Contact him” button. This will open a new window in your email client with the recipient address and subject already filled. This can be necessary if a student has particularly bad results or shows much less interest and involvement in the course. It motivates teachers to be proactive and take the lead to find a solution instead of waiting for the end of the semester and only seeing the student failure.

As we can see, you can click on any quiz to have more information about the quiz but most importantly see exactly what the student answered. It is possible to see his answers, what he answered, what was the correct answer, etc.

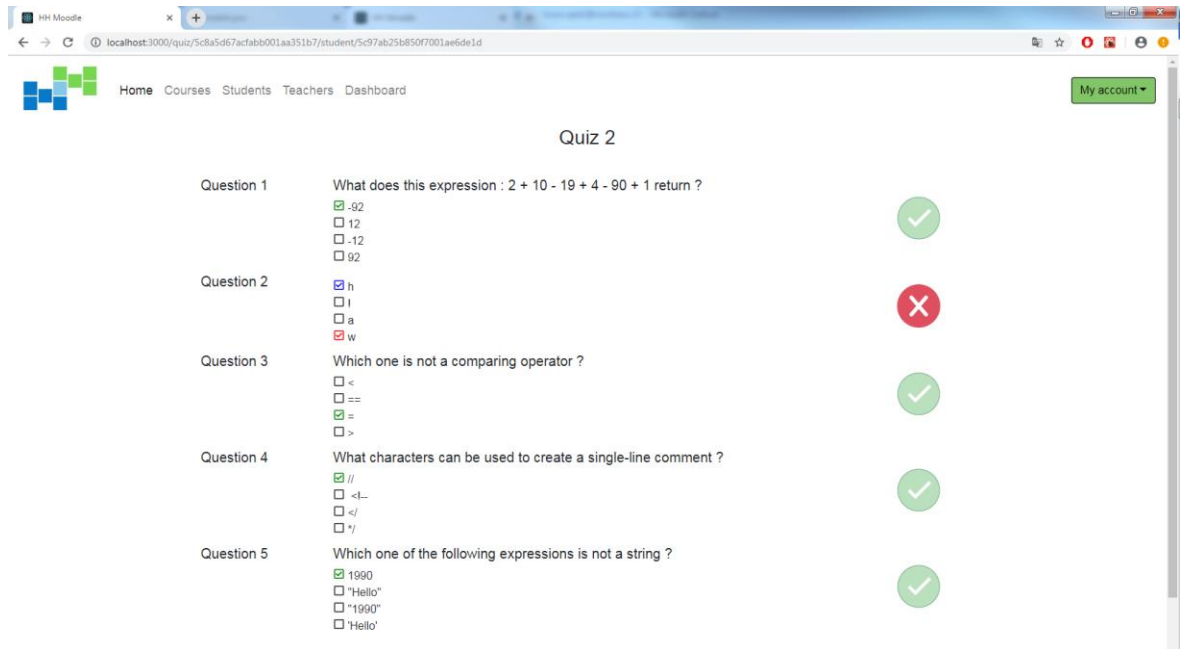


Figure 32 : Example of student's quiz result

Another option for the teacher after he created a quiz is to see some statistics about it. At any time, teachers can check the quizzes they created and obtain some immediate stats about it. For each question, they can see the number of persons that answered the question, the number of correct answers, incorrect answers and the percentage of correct answers. Therefore, teachers can analyse their quizzes and modify or adapt them if needed (if a question is too complicated for example).

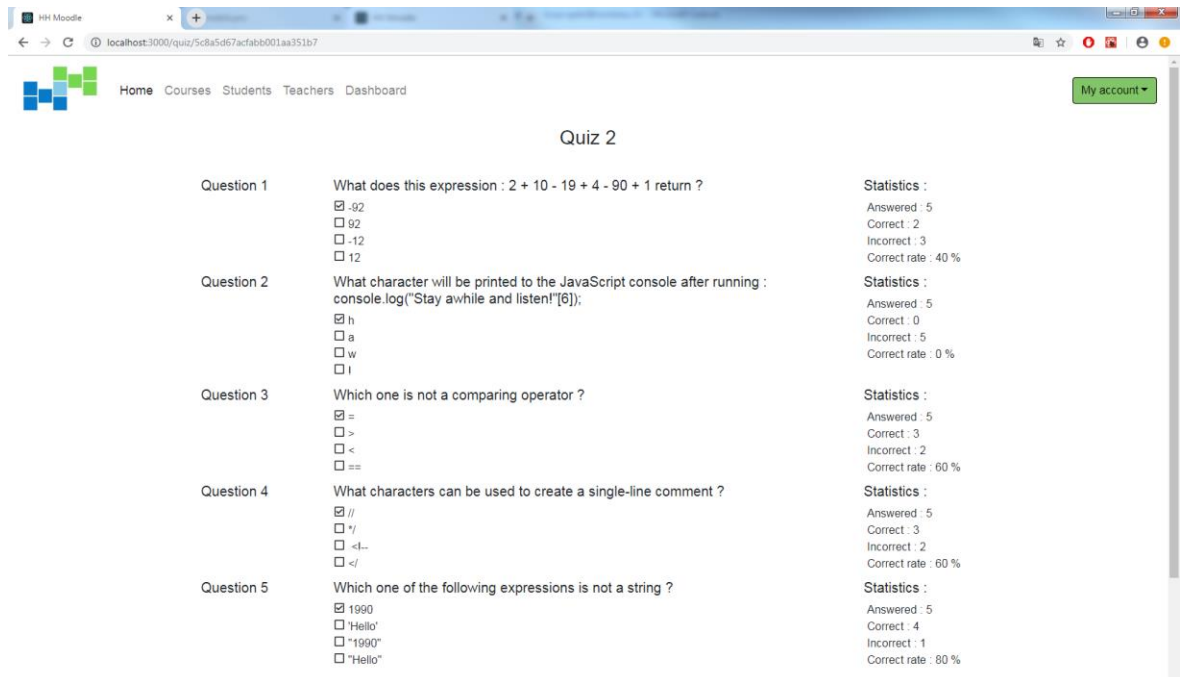


Figure 33 : Quiz with statistics

In the figure above, we can see that 0% of the students got the second question right. Maybe it is just a coincidence but maybe the question is too hard or maybe the answer is not in the resources. If that is the case, the teacher can modify the question. That is up to the teacher to decide but these stats definitely give him a good overview of his quizzes.

## 5 Implementation

The implementation of the system is based on a three-tier architecture. There is the front end which corresponds to the mobile application for the students and the website for the teachers. From the mobile application, everything that the student does such as taking a quiz or reading a lecture is automatically saved and data is sent to the server via an application programming interface (API). This data is stored in a NoSQL database. For this architecture, we used a MongoDB database.

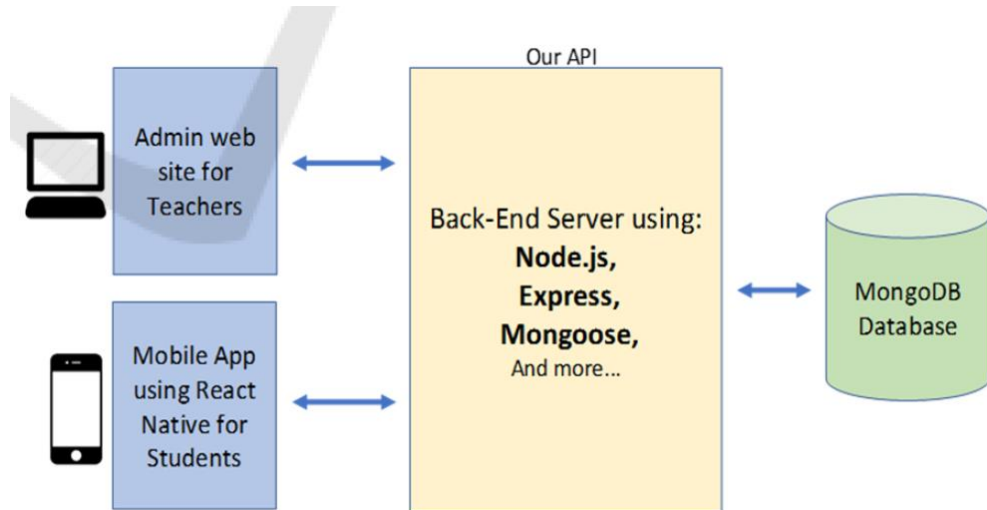


Figure 34 : Full architecture designed by Dr. Dirin and Laine (Dirin & Laine, 2018)

### 5.1 Mobile application

The mobile application is written using the React Native Framework (Danielsson, 2016).

We choose to use this framework for multiple reasons. First of all, an application developed using this framework can be easily deployed on both iOS and Android devices and therefore covering all smartphones' operating systems. The second reason is that React Native is built from components that are reusable and allow the mobile application to render natively. Plus, the administrative website for teachers is written in React.js as well and therefore, the same logic could be applied to the both of them. Finally, it is one of the best mobile frameworks and its popularity keeps growing amongst the developer.

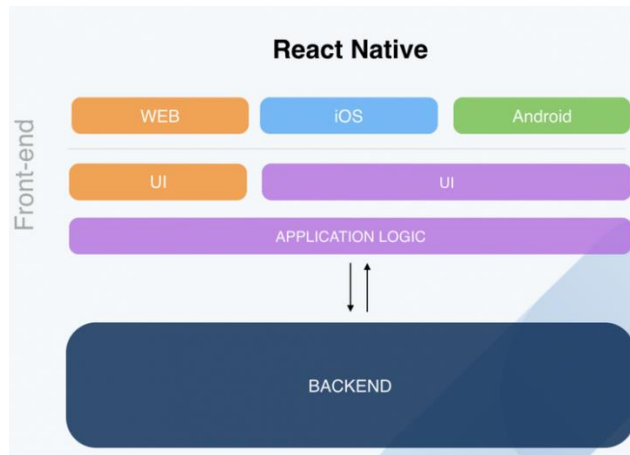


Figure 35 : Structure of React Native (5 key advantages of React Native, 2017)

In order to conduct our research, we deployed the application for iOS and Android. We generated a .apk file for Android and a .ipa file for iOS. These are the files needed to install our application on smartphones.

## 5.2 Website

The website is written using HTML and CSS and uses the most popular web framework which is ReactJS. ReactJS is a JavaScript library for creating user interfaces. It corresponds to the view layer of the MVC (Model, View, Controller) architecture.

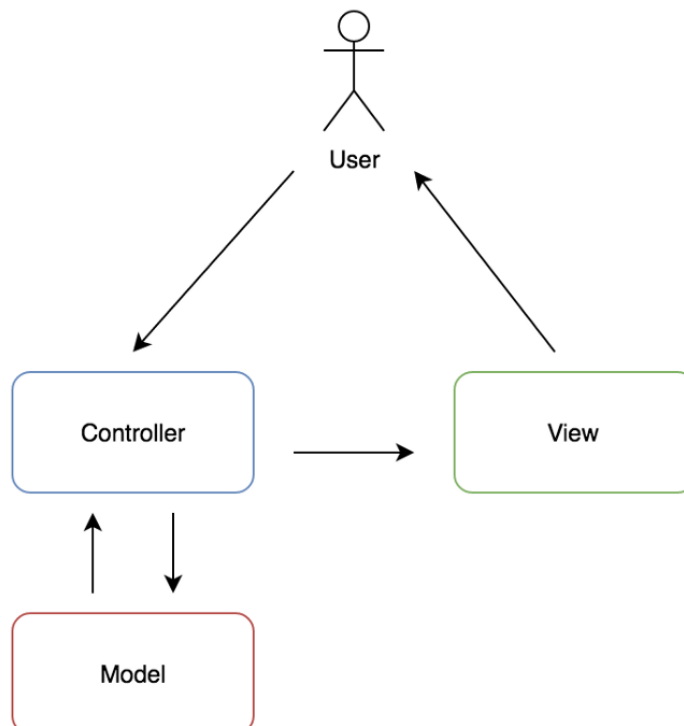


Figure 36 : MVC Architecture (Deutsch, 2017)

ReactJS uses the speed of JavaScript to render webpages dynamically (Thinkwik, 2017). In addition to that, we used Bootstrap to make pages responsive on both computer browsers and mobile devices. Bootstrap is the world's most popular front-end component library in the web development.

The website uses components to render pages. Components are created once and can be used many times. For example, we created the component "Table" that we used multiple times to render students, teachers and courses list in it.

```
83 render() {
84
85 //Initiate the columns that are passed to the table component
86 const adminColumns = [{dataField: 'firstName'...}, {
91   dataField: 'lastName',
92   text: 'Last name',
93   sort: true
94 }, {dataField: 'number'...},{dataField: 'email'...},{dataField: '_id'...}, {dataField: '_id'...}, {dataField: '_id'...}];
115
116 const columns = [{dataField: 'firstName'...}, {dataField: 'lastName'...}, {
125   dataField: 'number',
126   text: 'Number',
127   sort: true
128 }, {dataField: 'email'...},{dataField: '_id'...}];
137
138 //Verify if the user connected is an admin
139 if(sessionStorage.getItem( key: 'role')=="admin"){
140   return (
141     <div className="col-10 mx-auto">
142       <h3 className="titleMarginTop text-left">Students</h3>
143       <Table data={this.state.students}
144         columns={adminColumns}
145         id="number"
146         sort="lastName"
147         search="a student"/>
148       <Link className="btn btn-custom float-left" to="/student/add">
149         Add a student
150       </Link>
151       <NotificationContainer/>
152     </div>
153   );
154 }else{
155   return (
```

Figure 37 : Sample of render method of the "Students" component

The render method is the only required method in a class component in React. It returns React elements created via JSX such as the HTML <div> element or any other custom element <CustomElement>. As we can see in the figure above, the render method of the "Students" React element returns some HTML elements such as <div> or <h3> but also a <Table> element.

The `<Table>` element is a custom React element that can be called many times and has its own render method.

```
//Component class that renders a fully customized and responsive table using 'react-bootstrap-table2'
export class Table extends Component {
  render() {
    //Define the default sorting options (col & asc)
    const defaultSorted = [{order: 'asc'...}];
    //Define pagination options
    const customTotal = (from, to, size) => (
      <span/>
    );
    //Define the options of the table (nb of rows/pagination, etc)
    const options = {firstPageText: '<<'...};

    //Return a custom table with a search bar
    return (
      <ToolkitProvider
        keyField={this.props.id}
        data={this.props.data}
        columns={this.props.columns}
        noDataIndication={'no results found'}
        search
      >{props => (
        <div>
          <br/><SearchBar {...props.searchProps}
            className="custom-search-field"
            delay={1000}
            placeholder={`Search ${this.props.search}`}/><br/>
          <BootstrapTable defaultSorted={defaultSorted} {...props.baseProps}
            pagination={paginationFactory(options)}/>
        </div>
      )}
    </ToolkitProvider>
  )
}
```

Figure 38 : Custom React element "Table"

We chose to use ReactJS as it is extremely convenient and popular nowadays and it perfectly matches with our needs.

### 5.3 Back-end server

The back end corresponds to the server offering the API. The server uses Node.js which is the best and most used open-source, cross-platform runtime environment that executes JavaScript code outside of a browser. Nowadays, almost every server uses Node.js.



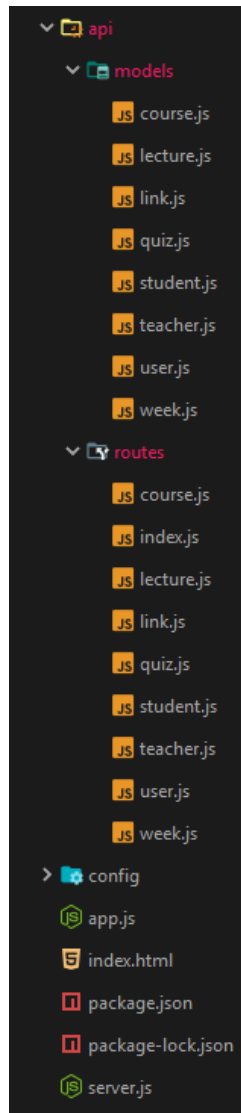


Figure 39 : Structure of the back-end server

Along with Node.js, we used the web application framework Express which is also a free and open-source software commonly used in back-end development. Express is typically designed for building web applications and APIs. It is also the backend component of the MERN stack.

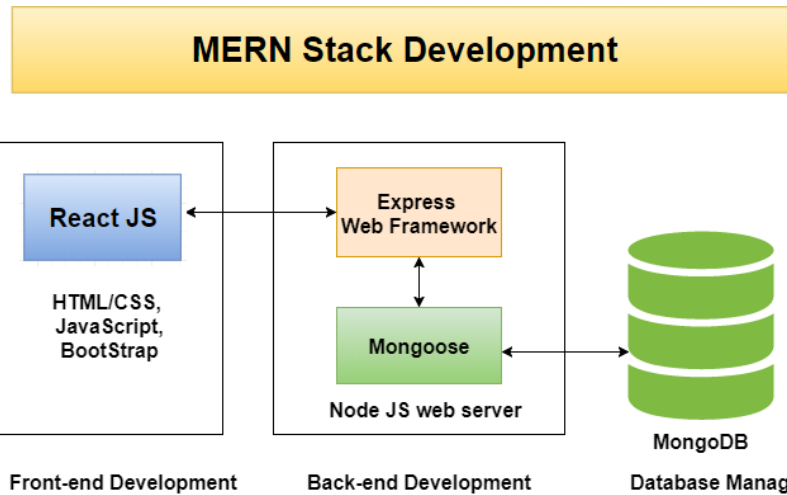


Figure 40 : Graphic of the MERN Stack components communication (Roy, 2018)

In our API folder, we can find models and routes. Models are used to describe the schema and methods of our objects.

```

const mongoose = require('mongoose');
const passwordHash = require('password-hash');
const jwt = require('jwt-simple');
const config = require('../../config/config');

//User schema
const userSchema = mongoose.Schema({
  _id: mongoose.Schema.Types.ObjectId,
  firstName: String,
  lastName: String,
  email: {
    type: String,
    required: true,
  },
  password: {
    type: String,
    required: true,
  },
  role : String,
});

userSchema.methods = {
  authenticate: function (password) {
    return passwordHash.verify(password, this.password);
  },
  getToken: function () {
    return jwt.encode(this, config.secret);
  }
}

module.exports = mongoose.model( name: 'User', userSchema);

```

Figure 41 : User model in the back-end server

In addition to that, we use Mongoose. Mongoose is an Object Data Modelling (ODM) library for MongoDB database and Node.js server. The role of Mongoose is to manage relationships between data. It also provides schema validation and is mainly used to translate between objects in code (here JavaScript) and the representation of those objects in the MongoDB database.

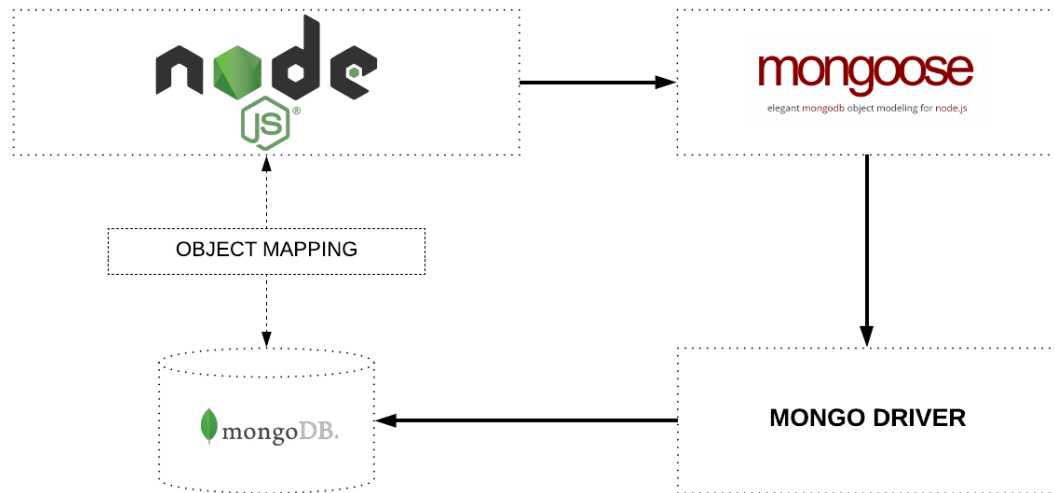


Figure 42 : Object mapping between Node.js and MongoDB via Mongoose

The routes folder contains all the routes of our API. For each object, we created routes using a specific HTTP method (GET, POST, PATCH, DELETE, etc.) (Mozilla, 2019).

If we want to get information about a specific user, we have to ask our server. To do that, we create a HTTP request from the client side (the mobile application or the website) and send it to the server. The HTTP request includes the request method, the request URI, headers and potentially body.

When the server receives the request, it performs some actions. In our example, if the server receives a request using the GET method on the specified URI with the correct header and the user id in the request parameters, it will try to find the corresponding user on the database using mongoose.

If the server finds a user, it will return it in the JSON format. Otherwise, it will return an error message.

```

24 //Get method to retrieve a user from the database by its id
25 router.get("/:userid", (req, res, next) => {
26
27     const id = req.params.userid;
28
29     User.findById(id)
30         .exec()
31         .then( onfulfilled: doc => {
32             console.log("From database " + doc);
33             res.status(200).json(doc);
34         })
35         .catch( onrejected: err => {
36             console.log(err);
37             res.status(500).json({error: err})
38         })
39     });
40
41 //Post method to save a new user in the database
42 router.post("/", (req, res, next) => {
43
44     const user = new User({
45         _id: new mongoose.Types.ObjectId(),
46         email: req.body.email,
47         password: passwordHash.generate(req.body.password),
48         role: req.body.role,
49     });
50
51     user
52         .save()
53         .then(result => {
54             console.log(result);
55             res.status(201).json({
56                 user: result
57             });
58         })

```

Figure 43 : Examples of routes from the server side

## **5.4 Database**

### **5.4.1 NoSQL**

The database is a NoSQL database. First of all, NoSQL is a concept. It is commonly interpreted as “not only SQL”. It gathers the family of all database management systems (DBMS) that want to overcome the constraints of the usually used relational database management systems (RDBMS) such as Oracle, MySQL, Microsoft SQL Server, PostgreSQL, etc.

NoSQL databases can handle a large variety of data including structured, semi-structured and unstructured data. NoSQL offers an alternative to the existing SQL language. It has a dynamic schema and therefore is extremely flexible.

NoSQL includes four different types of databases that offer a different way to represent the data. Each one of these offers advantage and disadvantages depending on the context where we want to use it. The first type is the key-value store, the second type is the wide-column store, the third one is the document-oriented store and finally there is the graph store. For our project, we decided to the document-oriented store so we won't get into details regarding the other types.

### **5.4.2 Document-oriented stores**

The document-oriented databases present the data as XML or JSON object. These allow to simply and quickly retrieve structured information in a hierarchical way. These work the same way as key-value stores except that the value is an object. Key-value stores are the simplest NoSQL databases. These allow storing the data as a key together with its value. This value can be a string, an integer or a serialized object like in our case. The most popular solutions are MongoDB, CouchDB and RavenDB.

Key	Value
CGHScore	{ "Title": "Chris Gayle Innings..." "UserId": "nrules" "Tags": ["IPL", "Cricket", "RCB"] "Body": "CG at his best..." }
RPForn	{ "Title": "Ricky P Innings..." "UserId": "nrules" "Tags": ["IPL", "Cricket", "MI"] "Body": "RP not his best..." } ...

Figure 44 : Records in a document-oriented database

For our project, we decided to use the MongoDB solution. MongoDB is one of the most popular and widely known NoSQL solutions. It is a document-oriented database that supports storing information in JSON format. It is open source and does not require a database administrator to administer. These features make MongoDB a very popular solution among developers and it is often used during the early stages of project development. MongoDB suits perfectly to companies that need to do large s of write operations and have big data-scale volumes to deal with. Also, it is commonly used when developing full stack application and is part of the popular MERN stack.

### 5.4.3 Collections and documents

Since our database isn't a relational database, we don't have the conventional database containing tables with columns and rows. Instead, we talk about collections and documents. Collections are to NoSQL databases what tables are to relational databases. Collections are analogous to tables while documents are analogous to records or rows in relational databases.

Since NoSQL databases are not some kinds of relational databases, there is no relation between tables or collections. It is difficult if not impossible to represent a MongoDB database as it is schema less by definition. However, it is possible to have a look at our database using some GUI (Graphical User Interface) tool such as MongoDB Compass. The figure below is the graphical representation of our database.

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size
courses	11	2.1 KB	22.8 KB	1	8.0 KB
lectures	44	242.9 B	10.4 KB	1	8.0 KB
links	49	278.6 B	13.2 KB	1	8.0 KB
objectlabs-system	1	112.0 B	112.0 B	1	8.0 KB
objectlabs-system.admin.collections	1	112.0 B	112.0 B	1	8.0 KB
quizzes	36	1.5 KB	52.4 KB	1	8.0 KB
students	52	6.5 KB	340.2 KB	2	16.0 KB
teachers	3	154.7 B	464.0 B	1	8.0 KB
users	56	240.0 B	13.1 KB	1	8.0 KB
weeks	60	231.5 B	13.8 KB	1	8.0 KB

Figure 45 : Database in MongoDB Compass

As you can see, we have multiple collections. We have a courses collection that gathers all courses and their respective info such as the name, the code, the scope, the level, the type, the objectives, the schedule, the path, etc. But most importantly, courses documents contain an array of weeks' id.

```

  _id: ObjectId("5c8f93e537b8a8001a60ec6e")
  weeksId: Array
    0: "5c8f941337b8a8001a60ec6f"
    1: "5c8f966b37b8a8001a60ec76"
    2: "5c8f9c1237b8a8001a60ec7b"
    3: "5c8fa05b37b8a8001a60ec81"
    4: "5c8fa33037b8a8001a60ec86"
  students: Array
    name: "Agile project management"
    code: "BUS1TF109"
    scope: "5 ECTS"
    timing: "1st Semester"
    language: "English"
    level: "Optional studies"
    type: "Elective"
    path: "Business"
    objectives: "The course familiarizes students to understand corporate IT developmen..."
    teacherId: "5c8a1a61f78a770019265864"
  schedule: Object
    day: "Wednesday"
    startHour: "1970-01-01T12:45:00.000Z"
    endHour: "1970-01-01T14:30:00.000Z"
    room: "3008"
  __v: 0

```

Figure 46 : Example of a course document

The « weeksId » array contains the “id” or identifier of the week documents. In the database, there is a weeks collection that regroup all weeks created by the different teachers. A week document is a document that contains 3 different arrays. The first array collects the id of the different related lectures while the second array collects the id of the different related external links or additional resources. Finally, the third array contains the id of the different quiz for that specific week.

In fact, the weeks collection is like a reference table that link the different documents in the different collections. Once we got the course, we have the weeks references and then we can get the lectures, links and quizzes references as well. With the id of the lecture for example, it is possible to retrieve it and obtain its content.



```

    _id: ObjectId("5c8b7a24acfabb001aa351e9")
  ✓ lecturesId: Array
    0: "5c8b7cc4acfabb001aa351ea"
    1: "5c8b7cdeacfabb001aa351eb"
  ✓ linksId: Array
    0: "5c8b7d09acfabb001aa351ec"
    1: "5c8b7d2aacfabb001aa351ed"
    2: "5c8b7d3eacfabb001aa351ee"
  ✓ quizzesId: Array
    0: "5c962664b850f7001ae63012"
  no: 4
  __v: 0

```

Figure 47 : Week document with its arrays

In our database, we also have 3 different collections for the users. The first one is the students collection. All students' documents are stored in this collection. As for the students, we have a teachers collection. All teachers are stored in this collection as well. It is pretty similar to the students collection except that there are fewer details and information since we don't collect data for the teachers.

```

_id: ObjectId("5bebed87e4e0e774e4eb6981")
firstName: "Amir"
lastName: "Dirin"
email: "amir.dirin@haaga-helia.fi"
__v: 0

```

```

_id: ObjectId("5c0c2b227e0f7236ac6060e4")
firstName: "Kari"
lastName: "Silpiö"
email: "kari.silpio@haaga-helia.fi"
__v: 0

```

```

_id: ObjectId("5c8a1a61f78a770019265864")
firstName: "Bryan"
lastName: "Spahr"
email: "bryan.spahr@myy.haaga-helia.fi"
__v: 0

```

Figure 48 : Overview of the teachers collection

Finally, we have a users collection that regroups all users which includes students and teachers. The difference between the users collection and the students and teachers collection is that the last two collections are used to store details about a student or a teacher while the users collection is used for the authentication.

The user documents are the same for the teacher and the student in the contrary of the student and teacher documents. The users collection contains user documents that gather basic information such as the email, the password and the role.

There are 3 possibilities of roles. The user role can be student which is reserved for students, teacher which is reserved for teachers and finally admin which is used by administrators. The major difference is between the student and the teacher role. A student won't be able to log in the website because they are only students. On the contrary, a teacher will be able to log in because he needs to manage his courses and can access the list of students. The admin role is similar to the teacher role except that he adds some functionalities on the website such as creating a new course, adding a new student, deleting teachers, etc. The administrator has all rights.

---

```
_id: ObjectId("5c8a193ef78a770019265863")
email: "1"
password: "sha1$5cb6d452$1$42161a2ca95d894b144f8f8ce8c38fd3e520f3c9"
role: "admin"
__v: 0
```

---

```
_id: ObjectId("5c8a20e51436bb001952f3d5")
email: "bryan.spahr@myy.haaga-helia.fi"
password: "sha1$81e0d4ee$1$67a05c31dca52b961ab47f4822c31a584e6b98cd"
role: "teacher"
__v: 0
```

---

```
_id: ObjectId("5c8cbfb9c50d210018252a16")
email: "Ya"
password: "sha1$fe5f6d50$1$1ac034bdfd569c5727bb1091fc62ead908bef994"
role: "student"
__v: 0
```

---

```
_id: ObjectId("5c90a5cf37b8a8001a60ec9c")
email: "kimmie.jax@myy.haaga-helia.fi"
password: "sha1$84c47296$1$d060089f282b567b2d38c73a2fe37f98e7dda9ad"
role: "student"
__v: 0
```

---

Figure 49 : User documents

```

API.login(_send)
  .then(res => res.json())
  .then( onfulfilled: data => {
    console.log(data)
    if (data.token != null) {
      sessionStorage.setItem('token', data.token);
      sessionStorage.setItem('id', data.user._id);
      sessionStorage.setItem('role', data.user.role)

      if (data.user.role == "teacher") {
        API.getTeacherByEmail(this.state.email)
          .then((data) => {
            sessionStorage.setItem('teacherId', data.data._id);
          })
      }
      window.location = "/dashboard"
    }
  })
  .catch( onrejected: (error) => {
    console.error(error);
  })

```

Figure 50 : The role of the user is stored in the session storage after log in

```

if(sessionStorage.getItem( key: 'role') == "admin"){
  return (
    <div className="col-10 mx-auto">
      <h3 className="titleMarginTop text-left">Teachers</h3>
      <Table data={this.state.teachers}
        columns={adminColumns}
        id="email"
        sort="lastName"
        search="a teacher"/>
      <Link className="btn btn-custom float-left" to="/teacher/add">
        Add a teacher
      </Link>
      <NotificationContainer/>
    </div>
  );
}else{

```

Figure 51 : The role of the user is retrieved from the session storage

## 5.5 Global architecture

Together, all these components form what is called a MERN stack. MERN is the acronym of MongoDB, Express.js, React and Node.js. The MERN stack is a totally free and open-source JavaScript software stack for building dynamic web sites and applications. All its components support programs that are written in JavaScript. The advantage of it is that all MERN applications can be written in one language only (JavaScript here) for both server side and client side (our mobile application) execution environments. MERN is a simple and efficient way to put all these components together in order to create a full stack application. It is also the most modern and popular way to do it.

The student interacts only with the mobile application which communicates with the server that communicates with the database. All the components communicate between each other so it creates links between all parts. Below is a sequence diagram illustrating how it works in practice:

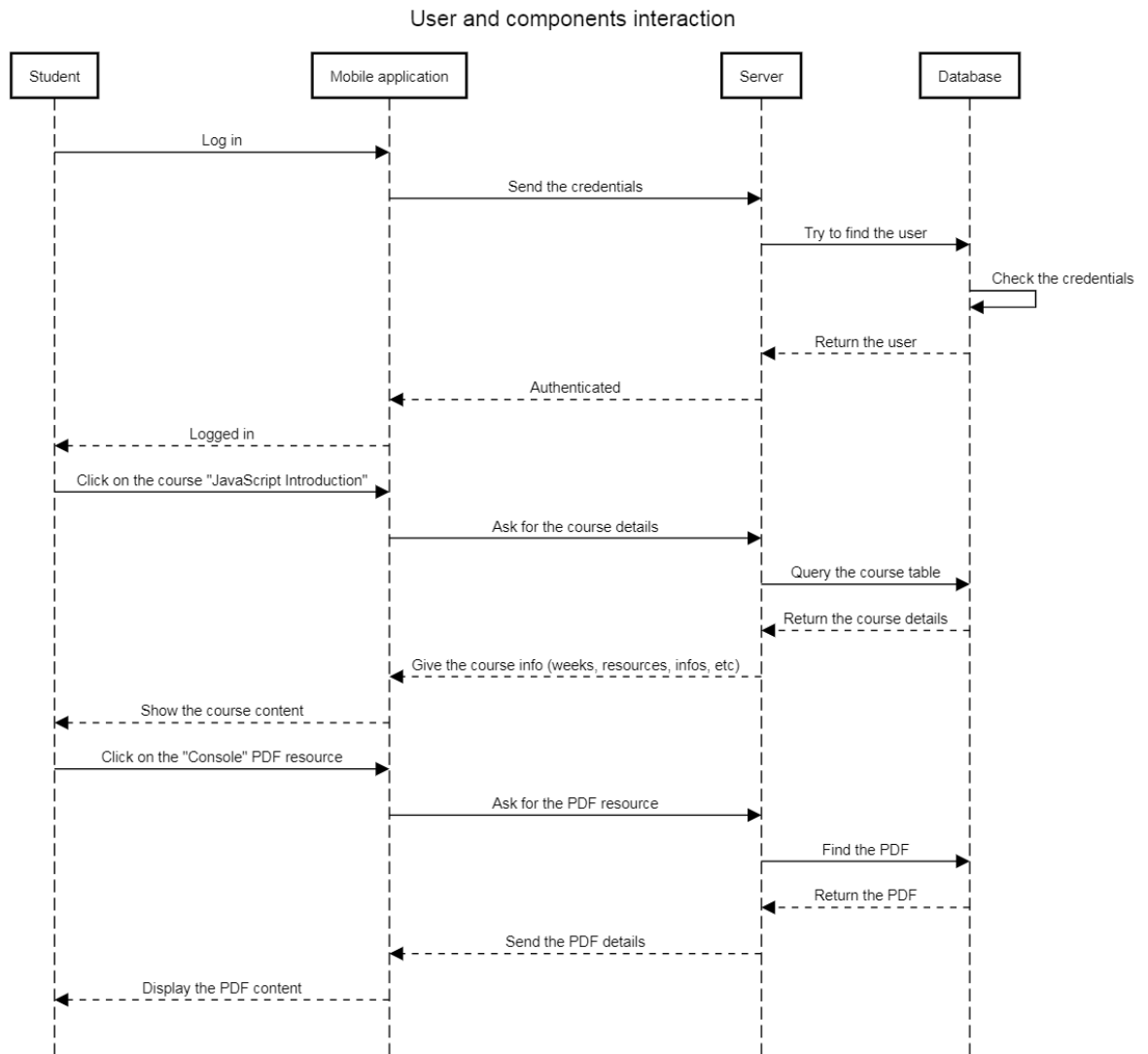


Figure 52 : Simple interaction between the user and the components

## 5.6 Student data

Every time students interact with the app, it creates data. Every action, its duration, its result (when taking quiz for example), everything is saved and stored. The data that students generate using the mobile application are thereby sent from the client side (mobile application) to the server side (back-end server) using the API. When the server receives it, it stores the data in the MongoDB database. Therefore, it is possible to track the steps of the user and its result.

For example, if a student logs in the application, browses his courses, chooses the “JavaScript Introduction” one and clicks on the PDF resource “Console”, it is possible to know it. If we look at the database, we can know when he logged in (the specific time including the day, the hour, the minute and even the second). We can also know the specific time he opened the PDF, we can retrieve the exact amount of time (to the nearest second) he

spent on the PDF. We can know the specific time he closed the application. It is also possible to know if the student opened a resource for the first time or if he already opened it before and was reviewing it.

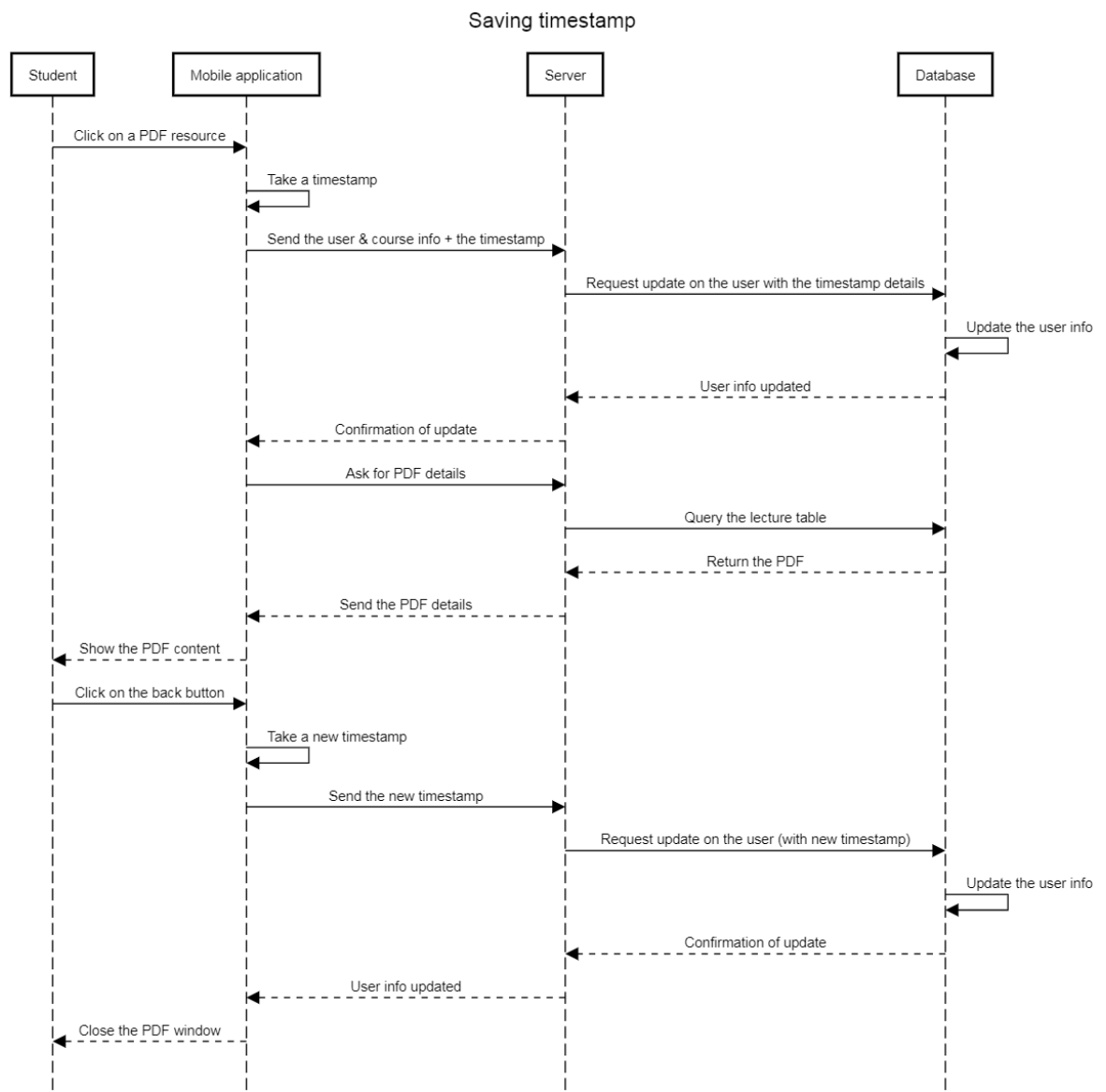


Figure 53 : Sequence diagram illustrating the saving of the timestamp

### 5.6.1 Course statistics

When the student opens the app and selects a course, a timestamp is automatically generated and the mobile application starts to time the duration of the navigation. Until the student closes the application, the stopwatch runs and keeps recording the time spent on the course. When the application is closed or when the student exits the course, it also generates a timestamp that is automatically saved in the database.

```
  _id: ObjectId("5c90a68f37b8a8001a60eca9")
  number: "a1702158"
  firstName: "Alexis"
  lastName: "Stacee"
  email: "alexis.stacee@my.haaga-helia.fi"
  password: "sha1$cd0cedfa$1$fafbedb821df02519914e15c56cb6ee748ae9fa5"
  courses: Array
    0: Object
      _id: ObjectId("5c972e4cb850f7001ae66e06")
      globalResults: Array
      quizResults: Array
      done: Array
      courseId: "5c8a1ba0f78a770019265865"
      path: "Programming"
      globalScore: 30.666666666666664
      timeSpent: 65.681
      percentage: 44
    1: Object
      _id: ObjectId("5c972e4cb850f7001ae66e04")
```

Figure 54 : Time spent by a student on a specific course

Therefore, it is possible to know exactly when a student started studying, when it stopped studying and the time he spent studying. With this information, we can see if a student spends too little or too much time on a course compared to other students and thereby find an explanation to its too bad or too good results. It is possible to create a relation between the time spent on a course and the results. Also it allows to see which students don't spend enough time and which are the most dedicated ones.

```

componentDidMount() {

  var openedAt = new Date();

  this.setState( state: {
    openedAt: openedAt
  })

  this.setState( state: {
    course: this.props.navigation.getParam('course', 'default'),
    student: this.props.navigation.getParam('student', 'default'),
  })
}

componentWillUnmount() {

  var closedAt = new Date();

  var timeSpent = (closedAt.getTime() - this.state.openedAt.getTime()) / 1000;

  this.saveTimeSpent(timeSpent);

  this.setState( state: {
    closedAt: closedAt,
    timeSpent: timeSpent
  })
}

```

Figure 55 : componentDidMount and componentWillUnmount methods of the Course component

ReactJS has some predefined methods that allow executing some code when a component is mounted and when it is unmounted. We used these methods and create a new Date object every time the component course is mounted. This object is initialized with the current time and date. This date object represents our timestamp. It is saved within the component until the component is unmounted. When this happens, another date object is always created with the current date and time and is saved within the component as well.

When the component is unmounted, we calculate the difference between the last date object (when the course was closed) and the first date object (when the course was opened).



## 5.6.2 Resource statistics

Every time the student interacts with the content of a course, everything is recorded the same way as the course stats. For example, when a student clicks on a lecture and starts reading it, the stopwatch starts and stops only when the student closed the app or the resource. Therefore, the time the student spent on every resource (lecture, video, image, website, etc.) is recorded, saved, sent to the server and stored in the database.

It is thereby possible to compare this information between the students. It is possible to know how much time a student spent on every resource and establish links between the most used resources and the results.

It is possible to know, for every resource, how many times it has been opened, how many times it has been opened by a unique user, how much time in total student spent on it, etc. We save and store a lot of information and stats regarding every resource.

```
_id: ObjectId("5c8a2ef41436bb001952f3da")
weekNo: 1
no: 3
title: "Console"
link: "https://pdfhost.io/v/RUTkqC7Ni_week1_consolepdf.pdf"
type: "lecture"
__v: 0
timeSpent: 84.16399999999999
uniqueTimesOpened: 6
timesOpened: 10
```

---

```
_id: ObjectId("5c8a5bcaacfabb001aa351b3")
weekNo: 2
no: 1
title: "Data Types"
link: "https://pdfhost.io/v/dRA@qOSSX_week2_dataTypes_variablespdf.pdf"
type: "lecture"
__v: 0
timeSpent: 44.95299999999999
timesOpened: 8
uniqueTimesOpened: 6
```

Figure 56 : Examples of lecture's stats

## 5.6.3 Quiz statistics

Quizzes' data are saved and stored on multiple levels. Firstly, when a student takes a quiz, his answers are saved and stored with his profile. Therefore, it is possible for a student to re-click on any quiz and see his results. He can see what he answered, if this was

correct or not and if not, what was the correct answer so he can learn from his errors. He can retrieve this information any time and as often as he wants.

Secondly, whenever a user answers a question, the result (correct or incorrect) is saved and stored alongside the question. Thereby, it is possible to know, for every quiz and every question, how many students answered correctly and how many students answered with an incorrect answer. Knowing that, the teacher could adapt his quiz, know if a question was too hard or too easy and have detailed statistics for each question.

Finally, like every resource, quizzes save and store the number of times it is opened and how much time student spent on it. With this information, it is possible to know the time a student spent on a quiz, the time all students spent on a quiz, the number of students that took the quiz and calculate the average time a student spends on a quiz for example. Knowing the average time, the teacher can detect students that didn't spend enough time to answer the questions or students that were too long to answer. There are multiple way of interpreting the statistics.

```
  _id: ObjectId("5c9623bfb850f7001ae6300e")
  weekNo: 3
  no: 1
  title: "Quiz 3"
  type: "quiz"
  questions: Array
    0: Object
      incorrectAnswers: Array
        0: "Define"
        1: "Prototype"
        2: "Test"
      _id: ObjectId("5c9623bfb850f7001ae63011")
      question: "which one is NOT a phase of the Design Thinking ?"
      correctAnswer: "Draw"
      nbCorrect: 0
      nbIncorrect: 4
    1: Object
    2: Object
  __v: 0
  timeSpent: 28.64
```

Figure 57 : Example of a quiz document in the database

## 5.7 Recommendation

One of the goals of this project is to provide students recommendation regarding their studies. As discussed earlier, before each semester starts, students have to choose courses in a very large list. In order to help them selecting the best courses that will suit

them the best and where they will be the most performant, the mobile application will provide students a study path recommendation.

For example, if a student performs particularly well in programming language courses such as Java or JavaScript but has poor result in design courses such as Photoshop or UX design, the mobile application will recommend him to choose courses from the “programming” path and will help him make choices. This applies for every semester.

The recommendation provided by the mobile application is based on the student’s data saw earlier in the document. The application will use artificial intelligence to retrieve the student’s information stored in the database, analyse it and make recommendation.

### **5.7.1 Data mining**

The data mining process represents the extraction of knowledge from a very important amount of data. The goal of data mining is to build models from the data. It tries to find interesting structures or patterns based on some predefined criteria in order to extract knowledge from it.

In our project, we collect all kind of data that we store in the database. When we want to make a recommendation, we retrieve these data and filter them to keep only the ones that interest us for our decision. In our case, we want to know for a student: the time he spent on a course, the number of resources he opened, the number of time he opened it, the time he spent on each resource, his individual result to each quiz and his the average result to these quizzes. We want to know these data for each course.

```

_id: ObjectId("5c90a68f37b8a8001a60eca9")
number: "a1702158"
firstName: "Alexis"
lastName: "Stacee"
email: "alexis.stacee@myy.haaga-helia.fi"
password: "sha1$cd0cedfa$1$fafbedb821df02519914e15c56cb6ee748ae9fa5"
courses: Array
  0: Object
    _id: ObjectId("5c972e4cb850f7001ae66e06")
    globalResults: Array
    quizResults: Array
    done: Array
    courseId: "5c8a1ba0f78a770019265865"
    path: "Programming"
    globalScore: 30.666666666666664
    timeSpent: 65.681
    percentage: 44
  1: Object
    _id: ObjectId("5c972e4cb850f7001ae66e04")
    globalResults: Array
    quizResults: Array
    done: Array
    courseId: "5c8b7036acfabb001aa351d3"
    path: "Design"
    globalScore: 41.666666666666666
    timeSpent: 47.745
    percentage: 41
  2: Object
  __v: 0

```

Figure 58 : Example of student data stored in the database

## 5.7.2 Machine learning

The machine learning is a field of study from the artificial intelligence (Wikipedia, 2019). It relies on statistical approaches in order to give computers the ability to learn from data to improve their performance and ability to solve tasks. Machine learning includes the conception, the analysis, the development and the implementation of such methods. The machine learning generally includes two phases. The first one consists of estimating a model from available data. This phase is also known as the observation phase. It is a training phase that is realized beforehand and tries to train the computer. The second step of machine learning corresponds to the implementation or deployment. Once the model is defined, new data can be submitted and the computer will be able to realize the tasks that are asked to it.

In our project, when we get the information we want from data mining, we can define some rules such as the most time the student spent on the course, the most interested he is by that course and thereby that study path. We can also imagine that the best results a student has in a course, the most performant he is and the most suitable the course is for him. Once we gathered these data, we use a formula that takes into account the data for each category and gives it a certain weighting according to the rules we have defined.

In our project, we decided to give a major importance to the quizzes results as it is a very good indicator of the student ability to pass in that course. Quizzes are by definition a way of evaluation the student knowledge. If a student appreciates the subject he will tend to be more involved in the course, study more and be more curious about it. He will also be more focused and the theory will be assimilated better. All these factors will play a role in the student results. Thereby, we can reckon that the quiz result of a student is a good representation of the student interest and involvement in a course which are the key to success.

On the contrary, bad results to quizzes can illustrate either a lack of involvement or a lack of interest by the student if not both. We reckon that a student having bad result to quizzes has more chance to fail the course than a student having good result to the same quizzes. Therefore, it is probably not a good idea for the student to take other courses in the same study path. Likely the student won't like the course or won't perform well and that is what we want to avoid.

However, a student can sometimes have bad results not because he doesn't like the course or because he is not studious enough but because he has more difficulty. A student can enjoy a subject, be extremely interested and nonetheless have poor result to quizzes. That is why we decided not only to take into account the quizzes results but also the involvement of the student in general. Even though the student may struggle in that study path, he probably will struggle even more in a study path that he doesn't like. We don't want to recommend a student to avoid courses he likes because he does not have the best results. On the contrary, we will encourage him to keep studying this subject and it is up to the teacher to pay particular attention to this student.

In our formula, we decided to weigh the involvement of the student by one third of the global score. The formula adds one third of the percentage of resources the student has opened to the quizzes average score. The result of this addition is considered as the "total" score of the student. We repeat these calculations and apply these rules and formula to each course. Once it's done, we compare the courses between each other by comparing this total score. In case of parity between the total score of different courses, we decide the best course by the total time the student has spent on that course. Finally, we rank the course by their total score.

```

getFavoritesCourses = (courses) => {

  const studentCourses = this.state.student.courses;

  var coursesTotal = [];

  for (var i = 0; i < courses.length; i++) {

    for (var z = 0; z < courses.length; z++) {

      if (courses[z].id == studentCourses[i].courseId) {
        var name = courses[z].name
      }
    }

    let c = {
      'name': name,
      'globalScore': studentCourses[i].globalScore,
      'percentage': studentCourses[i].percentage,
      'timeSpent': studentCourses[i].timeSpent,
      'path': studentCourses[i].path,
      'total': studentCourses[i].globalScore + (studentCourses[i].percentage / 3)
    };

    coursesTotal.push(c);

    if (i + 1 == studentCourses.length) {
      this.sortCourses(coursesTotal);
    }
  }
}

```

Figure 59 : Application of the formula

```

orderFavoritesCourses = (a, b) => {
  if (a.total > b.total)
    return -1;
  if (a.total < b.total)
    return 1;
  if (a.total = b.total) {
    if (a.timeSpent < b.timeSpent)
      return 1;
    else
      return -1;
  }
  return 0;
}

```

Figure 60 : Ordering the courses by their total score

Once the courses are ordered from the highest total score to the lowest total score, we sort these courses by their study path. Imagine that a student takes part in 5 different programming courses and 5 different design courses. Maybe the student can do well or very well in all design courses and extremely poorly in 4 of the 5 programming courses. The only exception is a programming course that is about a programming language that he already knows very well and thus has perfect quizzes results.

If we only take into consideration the total score of the best course and we decide to recommend its study path to the student, we would recommend him to take courses in the programming path even though it is not his favourite subject or where it is the best on average. That is why we decided, once the courses are ordered, to loop into them and allocate their respective study path some points based on the ranking of the course.

Once this is done, we divide the number of points for a path by the number of courses from that specific path in order to obtain an average number of points for this path. Then, we compare the different paths and order them by their total number of points on average. We apply the same method as for the courses and rank them from the highest number or points to the lowest number of points. Finally, the path with the highest number of points is the most suited path for the student and thereby the path that we recommend the student to choose his courses from for the next semester.

```

sortCourses = (courses) => {

    courses.sort(this.orderFavoritesCourses);

    var path = [
        {name: "Design", points: 0, nb: 0},
        {name: "Programming", points: 0, nb: 0},
        {name: "Business", points: 0, nb: 0},
        {name: "Technology", points: 0, nb: 0},
    ];

    var points = [25, 18, 15, 12, 10, 8, 6, 4, 2, 1];

    for (let h = 0; h < 10 && h < courses.length; h++) {

        switch (courses[h].path) {
            case 'Design':
                path[0].points += points[h];
                path[0].nb += 1;
                break;
            case 'Programming':
                path[1].points += points[h];
                path[1].nb += 1;
                break;
            case 'Business':
                path[2].points += points[h];
                path[2].nb += 1;
                break;
            case 'Technology':
                path[3].points += points[h];
                path[3].nb += 1;
                break;
        }
    }
}

```

Figure 61 : Sorting of the courses by their study path

As you can see in the figure above, the number of points given to a study path decreases as the courses are examined. For example, the path of the courses with the highest total score will receive 25 points when the path of the second courses with the highest score receives 18 points and so on.



Once this is done, we loop into the different paths and divide their total number of points by the number of entries they received in order to obtain an average number of points. Finally, we sort the paths by their number of points and set the recommended path as the path with the highest number of points. The recommended path is the path that is most likely to suit the student.

```
for (let i = 0; i < path.length; i++) {
  path[i].points = path[i].points / path[i].nb;
}

path.sort( compareFn: (a, b) => (a.points < b.points) ? 1 : ((b.points < a.points) ? -1 : 0));

this.setState( state: {
  favoriteCourses: courses,
  recommendedPath: path[0].name
})
```

Figure 62 : Sorting the different paths to recommend one

All processes of machine learning and its different algorithms are applied directly in the mobile application. Therefore, all calculations leading to the recommendation are executed on the client side and are immediately available to the student.

No calculation or algorithm is executed by the server. The back-end server only sends and receives data. Its functions are limited to simple services. The artificial intelligence or machine intelligence leading to the recommendation of the path is used exclusively on the client side.

## **6 Result**

### **6.1 Study case**

The case study conducted on the 5 students of the Haaga-Helia University was subject to a report including the test plan, interview questions and answers, test cases and raw data from the testing.

Overall, the test was conclusive. All students admitted that they would like to use the application if it was available and reckoned that it was both interesting and useful for them. Some recommendations were made and ideas of improvements were given but the overall feeling was very positive and the users rated the app highly with an average score of 6.4 out of 7. The details of the grade are discussed below.

#### **6.1.1 Interview results**

When the users were asked to tell about the things that they found difficult using the application they answered that nothing was difficult. The majority of them also didn't feel any frustration using it.

After doing some predefined tasks, students were asked some questions regarding their experience and the most important and useful feature in the application. Three of them mentioned that the navigation was very clear and intuitive when two of them said that the major asset of the app was its list of courses and materials. They all agreed on the fact that the application was clearer when the courses were divided in weeks. They also noticed that the files were easy to find and they appreciated the progress bar. They judged the recommendations feature extremely interesting and useful. The fact that PDF files could be opened in the app and didn't need to be downloaded also delighted them.

They were then asked to give their general impression about the application. Two of them said that the application was nice overall when one said that the application could be really useful for his studies. The application was seen as even better than the actual Moodle website. However, one of the test users mentioned that the application could benefit a better design.

The things that stood out in the application were the clear user interface and the fact that there were weekly quizzes available. Also the whole course remains available throughout the entire duration of the studies so you can use the application at any time.

The test users found the application intuitive and very simple to use. They rated the ease of use of the application from 1 to 7. Two of them gave the maximal grade regarding the ease of use of the application when one gave the grade 6 and two others the grade 5 for an average grade of 6 out of 7, which is a very good result.

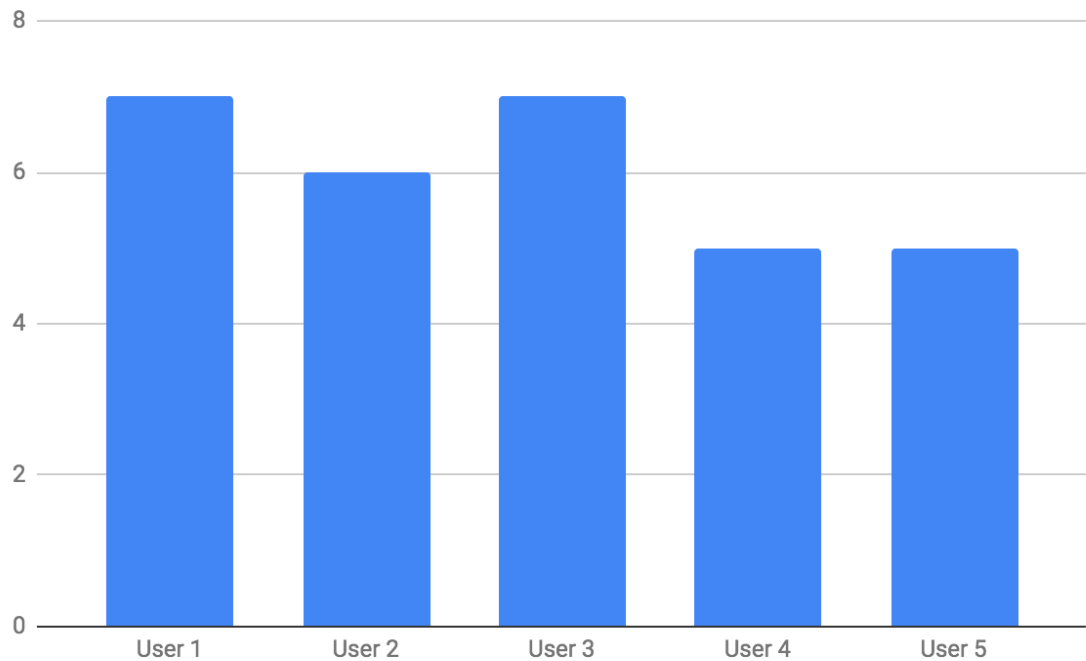


Figure 63 : Ease of use of the application (from 1 to 7)

All users found the application easy to use. They also made comments to support this score. They mentioned that the navigation and the user interface were extremely clear. The fact that weeks were nicely divided also played a role in the final grade. Finally, they liked the fact that icons were used to easily recognize and differentiate quizzes from PDF files for example.

Users also rated from 1 to 7 how likely they would be to use the application if it was available. Three of them answered without hesitation that they would definitely use it and gave the maximal grade (7). The user that gave the lowest grade (5) justified this score by admitting that he would use the application but not every day.

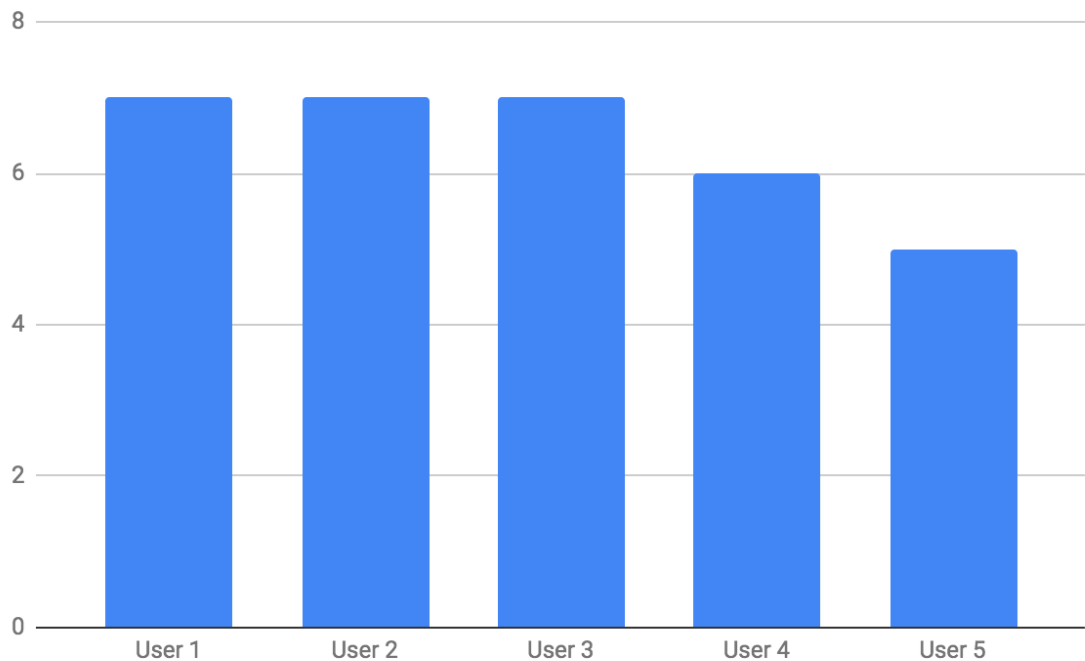


Figure 64 : How likely the test users are going to use the application (1-7)

Students made comments about why they would use the application. They liked that everything was in the same place and User 1 even found that this application was better than the similar ones he tried. In conclusion, with an average grade of 6.4 out of 7, students are extremely likely to use this application. This is a conclusive and motivating result as it is the end goal of this application.

### 6.1.2 User Experience ratings

Users were asked to rate some adjectives from 1 to 5 on how accurate it describes the application. The adjectives were the following: *interesting*, *fun*, *exciting*, *boring*, *frustrating* and *complicated*.

Interesting was the most accurate description of the app from the users' point of view. The adjective received an average grade of 4.3 out of 5. The fact that this adjective was the highest graded one proves that the idea behind the app was right. Students found that the prototype was well made since it raised their interest.

On a personal note, we are extremely proud that this adjective was designed as the most accurate one because it is the goal of the application. This project and its app aims to provide an educational tool for students that bring added value. We are delighted that this objective was met.

The following adjectives were fun and exciting with an average grade of 3.2 out of 5. However, students recognized that the application was not supposed to be fun and exciting and its main goal was to be educational. They found that the purpose of the application was reached.

Finally, the adjectives boring, frustrating and complicated got respectively the grades 1.8, 1.6 and 1.4 from which we can draw the conclusion that the prototype was a success to some extent at least. Test users didn't have negative feelings while using it. These bad grades here are a good thing and a good indicator for us.

### **6.1.3 Bugs and issues**

During the test phase, users didn't raise any bugs or issues. As expected, they found the app very easy and quick to use. They appreciated the user interface that they considered simple and clear. As mentioned previously on this paper, that was the goal of the application, to stay simple and straightforward in order to ease the study. Users admitted that, this way, they could better focus on their study work.

However, some students raised issues when using the application. The most problematic one concerned the progress bar. Users did not understand where they could see their progress in the application and there was a confusion between the progress bar and the progress in the quiz.

Also one user pointed out the fact that there was some kind of confusion between the main page and the dashboard page. He sometimes when to the pages where all courses are listed when he wanted to go to the "real" main page that is also the home page. The student admitted that there might be a short-term memory issue caused by the situation and stress. He ended up by recommending adding a fixed home button that would bring back the user to the home page.

### **6.1.4 Improvements**

Test users provided some recommendations regarding the application including cleaning up the user interface within the course overview so it would be simplified and more pleasant to the eyes. Students also found that the background picture within the course brought

confusion to the user and complexity to the user interface. They also thought that it would be easier to see and read the progress bar that way.

The second recommendations students provided was to find a way to make the down bar of the application more visible. They judged it hard to recognize. Overall, the visual design of the app could be improved as well. Another improvement that could be added but would not be crucial would be to add some link to the Moodle website and the possibility to get some hints while doing the quiz.

Finally, students would appreciate if there was a setting feature available to control some of the features in the application.

### **6.1.5 Conclusion**

The examiners concluded by saying that the application was a very successful product for its purpose and students joined their opinions. They highlighted the fact that the application enjoyed a large amount of content and useful features especially in the course overview module.

Personally, we are happy that the case study was a success. Not only did the tests go all well but the students liked the applications. That is the most important and encouraging point of this research. We wanted to determine if students could be interested in such an application and, of course, if this prototype was working properly. No bugs were revealed during this testing phase so the application is completely usable.

Overall, we are extremely happy with the results and with this report. We were flattered by the scores the application obtained and by the positive feedback we received. We also took notes of the suggestions and recommendations that were very pertinent. It gave us some important information that we needed to continue developing this app and improving the user experience. It is extremely helpful for us to have another point of view especially when it comes from students and students from our own school in addition to that.

We would like to warmly thank the students that accepted to participate in this case study and accepted to give their time for that. Also, we would like to thank them for their report that was extremely complete and well presented. They even furnished us the raw data of the testing phase. We also thank the examiners that took part in this case study. It definitely helped and motivated us. It was a necessary and extremely helpful experiment.

## 6.2 Test case

After creating the students, generating some random values and inserting everything in the database, we logged in the application with the accounts of the 25 students in order to copy the ranking order of the recommendation table.

Then, we compared these numbers with the correct numbers that we wrote down earlier and which depend on the values fabricated.

### 6.2.1 Example 1

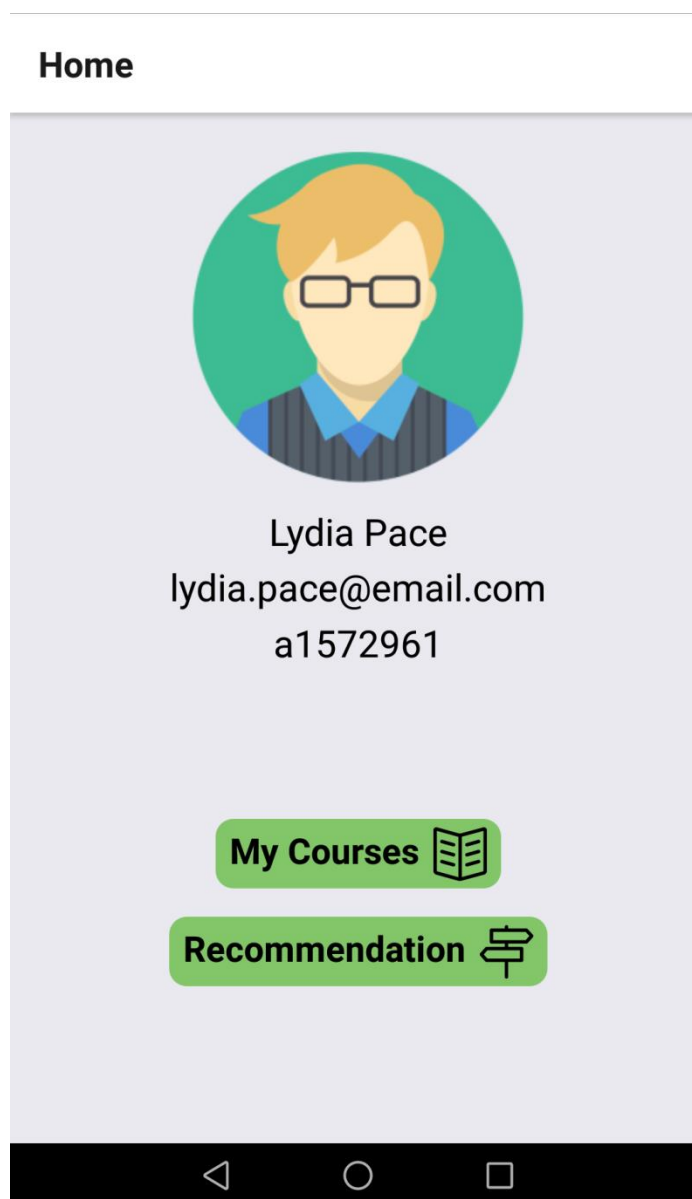


Figure 65 : Logged in as Lydia Pace (student 1)

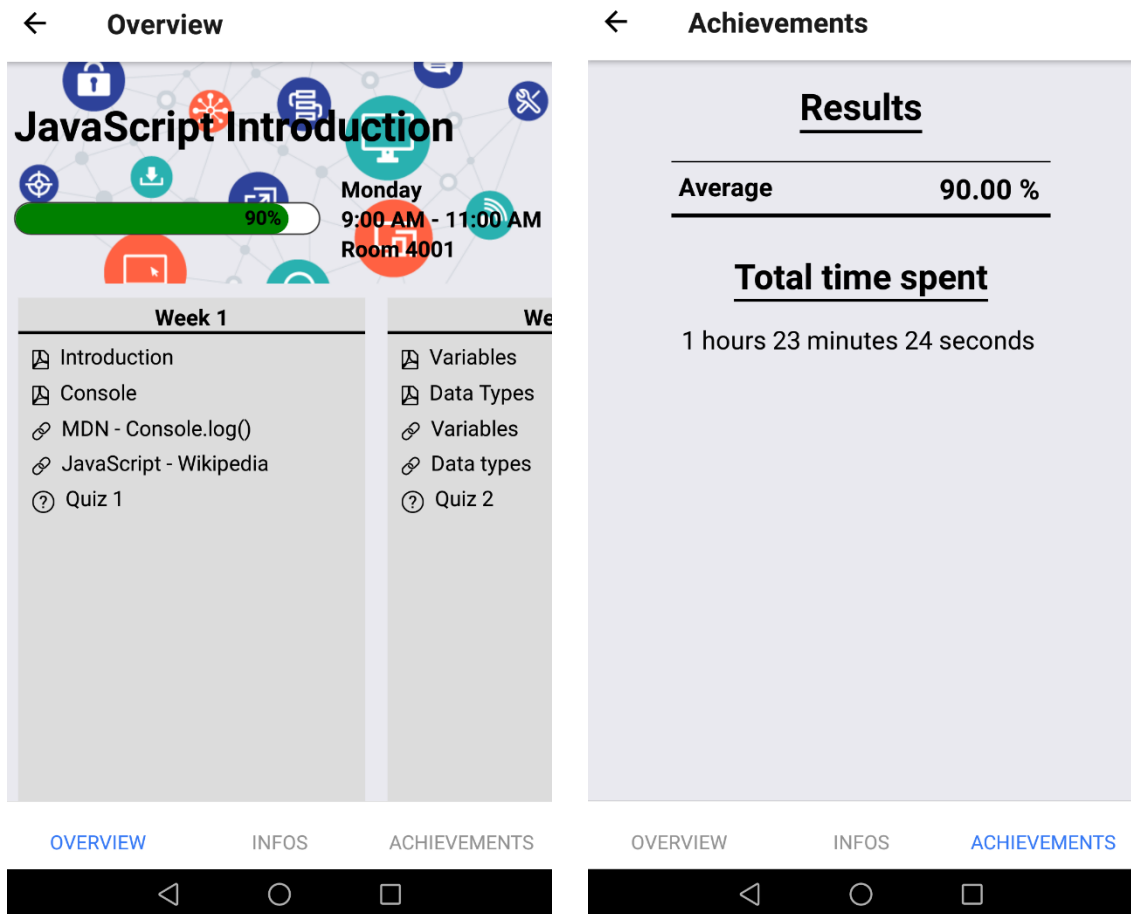


Figure 66 : Lydia's JavaScript Introduction course overview and achievements

As we can see on the figure above, for the JavaScript Introduction course, student 1 has done 90% of the course and spent 1 hour 23 minutes and 24 seconds which is approximately the 5000 seconds we entered in the database (+ 4 seconds the time we logged in).



Time to Decimal Calculator

hh : mm : ss  
1 : 23 : 24

Clear
Calculate

Answer:

= 1.3900 hours

= 83.40 minutes

= 5004 seconds

For the entered time  
01:23:24  
*1 hours 23 minutes 24 seconds*

Figure 67 : Time to seconds converter

D	F	G	H	I	J	K
Email	Course	Time spent (sec)	% Done	Global Score	Preference	Recommendation
lydia.pace@email.com	JavaScript	5000	90	90	1	
	UX Design	3500	80	70	2	
	Agile project management	1000	20	10	3	

Figure 68 : Values for student 1

Now, if we want to compare the system recommendation with what student 1 should have been recommended based on its values, we have to go on the recommendation table and look at the ranking order.



Figure 69 : Recommendation table on student 1 account

As we can see in the figure above, the ranking order of the recommended courses is similar to the ranking order we defined. In addition to that, the grade of the course recommendation gives us a very good idea of the recommendation accuracy. As we saw in the excel sheet, the difference between one course and another was pretty huge in terms of time spent, percentage done and global score so it is perfectly normal to have such a big gap in the grades.

However, you can still argue that it is easy and normal that the application recommends the same course as we did manually because of this huge difference between the values and that is why we also created students with extremely close values in order to test all the cases and possibilities.

## 6.2.2 Example 2













adam.branch@email.com	JavaScript	 5000	 80	 80	3
	UX Design	 5000	 82	 80	1
	Agile project management	 5000	 81	 80	2
luca.alexander@email.com	JavaScript	 6000	 80	 80	3
	UX Design	 6000	 80	 81	2
	Agile project management	 6000	 80	 82	1
lucia.clark@email.com	JavaScript	 5005	 50	 50	1
	UX Design	 5003	 50	 50	2
	Agile project management	 5001	 50	 50	3

Figure 70 : Students with extremely close values

In the figure above, you can see that we have 3 different students that have similar values for 2 criteria and extremely close values for 1 criteria.

For example, Adam Branch has the exact same time spent on the JavaScript course that on the UX Design and Agile project management course. He also has the exact same global score for these 3 courses which means that he is as well in all courses. The only difference here is that he opened more resources for the UX Design course than for the other courses by 1%. Obviously, this is a theoretical case because such a small difference would mean that he opened 1% fewer resources and therefore maybe 1 less PDF out of 100 PDF or more. However, it is interesting to see if, despite the similar time spent and global score and extremely close percentage of the course done, the system takes this small difference into account when making its recommendation.

If we log in the application with the credentials of Adam Branch and go to the recommendation table, we can see the following screen:

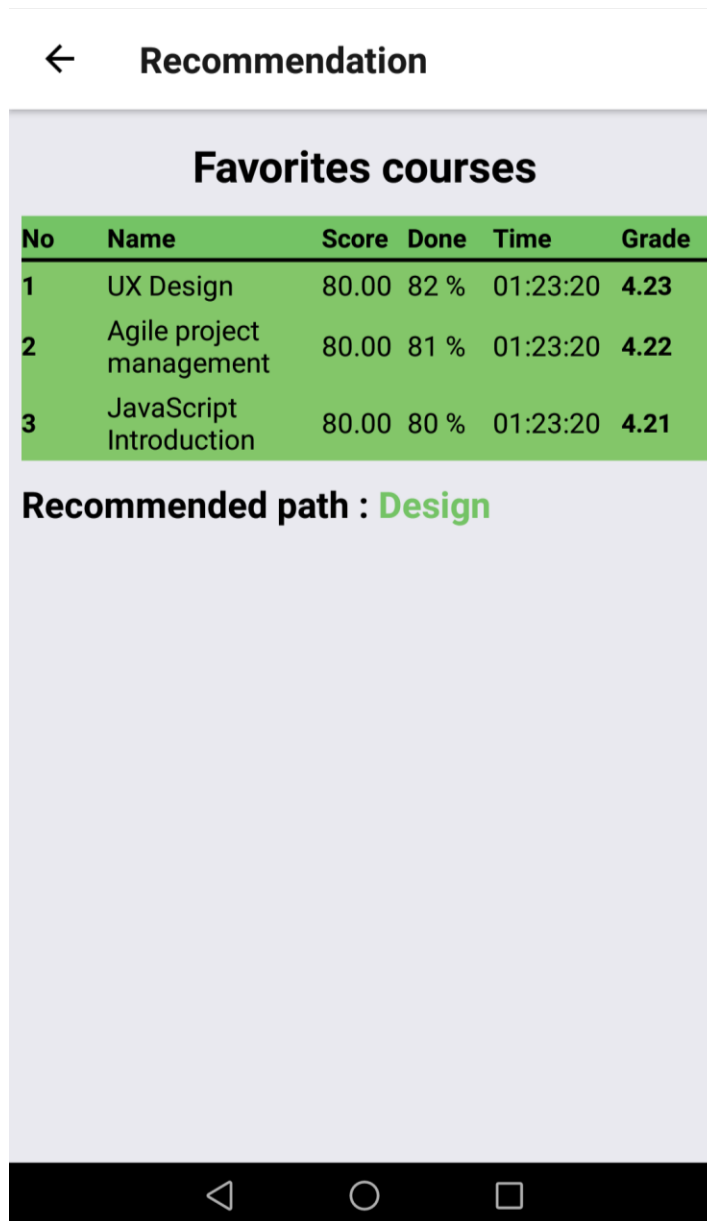


Figure 71 : Recommendation for Adam Branch

The above figure proves that, even with some similar and some close values, the system can distinguish the courses and still make an appropriate and correct recommendation based on these values. The ranking order provided by the mobile application is similar to the one we carefully calculated.

This example is the proof that the system reaches at the same conclusions as us and therefore, that the algorithm is totally correct. We can even see that the grades are different. That means that this very slight difference between just one of the three different values is passed on the grade and therefore is taken into consideration by the algorithm. The algorithm can differentiate close values. We now also have the proof that the system

takes everything into consideration when making its recommendation. All data, all values are used and weigh in the calculation.

### 6.2.3 Conclusion

As mentioned earlier, we created different students for this test case with different values. We tried to create all the possible scenarios in order to push the system in its limits and detect any kind of error.

For each student we created, we inserted its data in the database, logged in the application with its credentials and copied the recommendation table. We checked and compared all results for the 25 students.

Each student had 3 courses with 3 different values. At the end, the 3 courses were ranked from the most recommended one to the last recommended one. There were therefore 75 comparisons to make (25 students x 3 courses). The results can be seen in the figures below:

	A	B	F	G	H	I	J	K	L
1	Firstname	Lastname	Course	Time spent (sec)	% Done	Global Score	Preference	Recommendation	Correct
44	Malik	Sullivan	JavaScript	5000	50	100	1	1	Yes
45			UX Design	8000	60	10	3	3	Yes
46			Agile project management	2000	70	50	2	2	Yes

Figure 72 : Example of data and results for a single student

The full results can be seen on the following excel sheet:

	A	B	C	D	F	G	H	I	J	K	L
1	Firstnam	Lastname	Student num	Email	Course	Time spent [s]	% Don	Global Sci	Prefere	Recommend	Corre
14	Salma	Erickson	a1572961	salma.erickson@email.com	JavaScript	6000	80	80	1	1	Yes
15					UX Design	4000	80	80	2	2	Yes
16					Agile project manager	2000	80	80	3	3	Yes
17	Niokolos	Moran	a1572962	niokolos.moran@email.com	JavaScript	5000	90	90	1	1	Yes
18					UX Design	7000	70	70	2	2	Yes
19					Agile project manager	10000	10	10	3	3	Yes
20	Madeline	Mendez	a1572963	madeline.mendez@email.com	JavaScript	10000	95	95	1	1	Yes
21					UX Design	5000	100	50	2	2	Yes
22					Agile project manager	2000	20	30	3	3	Yes
23	Giselle	Harrison	a1572964	giselle.harrison@email.com	JavaScript	1000	20	30	3	3	Yes
24					UX Design	2000	40	60	2	2	Yes
25					Agile project manager	3000	90	90	1	1	Yes
26	Raegan	Miles	a1572965	raegan.miles@email.com	JavaScript	5000	50	50	3	3	Yes
27					UX Design	6000	60	60	2	2	Yes
28					Agile project manager	7000	70	70	1	1	Yes
29	Mia	Villegas	a1572966	mia.villegas@email.com	JavaScript	10000	50	80	3	3	Yes
30					UX Design	9000	90	80	2	2	Yes
31					Agile project manager	8000	95	85	1	1	Yes
32	Joey	Jensen	a1572970	joey.jensen@email.com	JavaScript	4000	80	50	2	2	Yes
33					UX Design	8000	90	70	1	1	Yes
34					Agile project manager	2000	20	35	3	3	Yes
35	Xiomara	Little	a1572971	xiomara.little@email.com	JavaScript	3000	20	50	3	3	Yes
36					UX Design	7500	85	80	1	1	Yes
37					Agile project manager	2500	55	70	2	2	Yes
38	Viviana	Fernandez	a1572972	viviana.fernandez@email.com	JavaScript	8000	30	90	1	1	Yes
39					UX Design	9000	60	60	2	2	Yes
40					Agile project manager	10000	50	40	3	3	Yes
41	German	Arnold	a1572973	german.arnold@email.com	JavaScript	4000	80	80	1	1	Yes
42					UX Design	7000	90	55	2	2	Yes
43					Agile project manager	9000	10	50	3	3	Yes
44	Malik	Sullivan	a1572974	malik.sullivan@email.com	JavaScript	5000	50	100	1	1	Yes
45					UX Design	8000	60	10	3	3	Yes
46					Agile project manager	2000	70	50	2	2	Yes
47	Eduardo	Briggs	a1572975	eduardo.briggs@email.com	JavaScript	5000	45	80	2	2	Yes
48					UX Design	8500	55	90	1	1	Yes
49					Agile project manager	10000	80	10	3	3	Yes
50	Dakota	Cox	a1572976	dakota.cox@email.com	JavaScript	1500	90	50	3	3	Yes
51					UX Design	5000	60	60	2	2	Yes
52					Agile project manager	2000	30	100	1	1	Yes
53	Charlee	Arellano	a1572977	charlee.arellano@email.com	JavaScript	5000	20	10	3	3	Yes
54					UX Design	2000	80	50	2	2	Yes
55					Agile project manager	8000	80	100	1	1	Yes

Figure 73 : Results of the test case

Out of these 75 comparisons, 75 were correct and not a single one was erroneous. The 25 students received the exact same recommendations from their mobile application that they would have received by a teacher or a school counsellor.

In conclusion, this test case proves that the application and its algorithm is properly working and doesn't make any judgement mistake. We can affirm that the recommendation engine built with the help of the artificial intelligence is totally reliable.

## 7 Discussion

When we began writing this paper we had a lot of open questions regarding artificial intelligence, students' learning activities and teachers teaching methods. We wanted to know if it was possible to gather these three different aspects in one unique project.

We live at a time where data has become extremely rich and powerful. Artificial intelligence, on the other hand, is more and more used as it is a proven method of adding value to anything. At the same time, the educational system has changed and teaching and learning methodologies have undergone a drastic change.

The challenge and the objective were to develop a modern platform that would take into considerations these changes and that would be in adequacy with its time. It is no longer possible to avoid using artificial intelligence and exploiting big data nowadays. Aware of that, the challenge was to bring a new and useful solution for everyone.

No similar project has been undertaken in the past. We had individual information about artificial intelligence, recommendation engine, studying mobility but nothing that combined all these aspects together.

After developing our project, testing it and conducting some research, we can conclude that there is an open window of opportunity in the use of artificial intelligence in the educational context. More encouraging, it definitely adds value for everyone. We revealed some curious but interesting points which deserve to be studied more in depth.

Today, we can affirm that it is possible to provide a study path recommendation to students with the help of artificial intelligence. We can prove that students' learning activities can be used as indicators for it and for teachers. We also demonstrated how teachers can adapt quickly and easily their course content based on students' learning outcomes. Also, we have the proof that a mobile application help and motivate students to learn and study.

## 8 Recommendation

Artificial intelligence is a large and wide area that requires time and investment to get the most out of it. However, it is also an extremely interesting and fascinating field. The use of AI has increased exponentially over the last few years in almost every field. There is no doubt anymore that it is a major asset in our environment and we have to work with it if we want to stay competitive and attractive. I can't imagine a business doing well without it nowadays. It can be profitable for both the user and the company. Everyone wins when using AI. The artificial intelligence and its sub-fields come with their advantages and bring a new world of possibilities and opportunities that we have to seize. We are at a turning point in our development and we have to get the most out of what technologies can bring to us.

In this project, we included 2 sub-fields of artificial intelligence that are data mining and machine learning. However, there is still a lot of other sub-fields that are equally interesting and beneficial for us. Similarly, we have barely touched these concepts so much their extent is great and their possibilities endless. It is already great to do what we have done but we can still do much more and we have to do it.

As we saw and discussed in this report, we could have used the same data again and again for different purposes and added other functionalities. There is no limit to what can be achieved.

I found this project more and more fascinating as the development progressed and I am convinced that the subject is fascinating and deserves to be deepened. The feedback from the study case confirmed that as the students were happy with the prototype and would have liked to use it on a daily basis. I sincerely think that if such a solution was implemented today, teachers and students would benefit from using it.

That is the reason why I encourage everyone reading this paper or interested in artificial intelligence applied in an educational context to pursue their idea and to develop new solutions like this one. I am fully convinced that one day students will study with their smartphones and that AI will help them make the right decisions for their future.

I regret the lack of interest and other researches regarding artificial intelligence in the educational context but I am confident this will change soon. I hope that people will get interested in similar solutions and that one day a school will build and use one. I am sure



that after seeing the benefits of it, other schools will quickly follow their steps and implement their own platform.

Finally, and regarding this project, I hope that it can serve Messrs. Dirin and Laine that imagined the development of such a solution in the first place. I also hope that someone will be interested in it and will want to continue or improve it. The source codes of the mobile application, the back-end server and the website are all available on public repositories on GitHub. It would be nice if the project continues to live and I would be extremely glad and honoured if someone uses it one day.

## 9 References

- 5 key advantages of React Native. (25. 04 2017). Noudettu osoitteesta ICAPPS:  
<https://www.icapps.com/blog/5-key-advantages-react-native>
- Adelina Moura, A. A. (2008). *Mobile Learning: Teaching and Learning with Mobile Phones and Podcasts*.
- Artificial Intelligence. (2018). Noudettu osoitteesta Wikipedia:  
[https://en.wikipedia.org/wiki/Artificial\\_intelligence](https://en.wikipedia.org/wiki/Artificial_intelligence)
- Baran, E. (2014). *A review of Research on Mobile Learning in Teacher Education*.  
Noudettu osoitteesta ResearchGate:  
[https://www.researchgate.net/publication/267337349\\_A\\_Review\\_of\\_Research\\_on\\_Mobile\\_Learning\\_in\\_Teacher\\_Education](https://www.researchgate.net/publication/267337349_A_Review_of_Research_on_Mobile_Learning_in_Teacher_Education)
- Claire O'Malley, G. V. (2005). *Guidelines for learning/teaching/tutoring in a mobile environment*.
- Cristóbal Romero, S. V. (2007). *Personalized Links Recommendation Based on Data Mining in Adaptive Educational Hypermedia Systems*.
- Danielsson, W. (2016). React Native application development: A comparison between native Android and React Native. *institutionen för datavetenskap*, (s. 20).
- Deutsch, D. (4. 2 2017). *Understanding MVC Architecture with React*. Noudettu osoitteesta Medium: <https://medium.com/of-all-things-tech-progress/understanding-mvc-architecture-with-react-6cd38e91fef>
- Dirin, A.;& Laine, T. (2018). Towards an Adaptive Study Management Platform: Freedom Through Personalization. *10th International Conference on Computer Supported Education, Funchal*, (s. 7). Madeira, Portugal.
- Eschweiler, S. (22. 12 2018). *The MERN Stack Tutorial - Building A React CRUD Application From Start To Finish*. Noudettu osoitteesta Medium:  
<https://medium.com/codingthesmartway-com-blog/the-mern-stack-tutorial-building-a-react-crud-application-from-start-to-finish-part-2-637f337e5d61>
- ETH Zürich. (2019). *Introduction to Machine Learning 2019*. Noudettu osoitteesta Learning & Adaptive Systems.
- Fernandez, A. (08. 10 2018). *Data Mining, explorer les données du Data Warehouse*. Noudettu osoitteesta Piloter.org: <https://www.piloter.org/business-intelligence/datamining.htm>
- Hamid Nemati, D. S. (2002). *Knowledge warehouse: an architectural integration of knowledge management, decision support, artificial intelligence and data warehousing*.

- Helwett Packard Enterprise. (2019). *Qu'est-ce que le machine learning ?* Noudettu osoitteesta HPE: <https://www.hpe.com/ch/fr/what-is/machine-learning.html>
- Hung-Chen Chen, A. L. (2001). *A music recommendation system based on music data grouping and user interests.*
- Jaggi Martin, U. R. (2019). Machine Learning. *Machine learning.* Lausanne: EPFL.
- Katakam, N. (07. 01 2019). *How Can We 'Design' An Intelligent Recommendation Engine?* Noudettu osoitteesta uxplanet: <https://uxplanet.org/how-can-we-design-an-intelligent-recommendation-engine-b9bb1db4d050>
- Keegan, D. (2005). *The incorporation of mobile learning into mainstream education and training.*
- L., B. (31. 01 2018). *Data Mining : qu'est ce que l'exploration de données ?* Noudettu osoitteesta Le Big Data: <https://www.lebigdata.fr/data-mining-definition-exemples>
- M., I. (06. 09 2017). *8 Reasons Your Ecommerce Store Needs a Mobile App.* Noudettu osoitteesta RubyGarage: <https://rubygarage.org/blog/benefits-of-mobile-app-for-ecommerce>
- Maruti Techlabs. (2019). *How do Recommendation Engines work? And What are the Benefits?* Noudettu osoitteesta Maruti Techlabs: <https://www.marutitech.com/recommendation-engine-benefits/>
- M-Learning.* (2018). Noudettu osoitteesta Wikipedia: <https://en.wikipedia.org/wiki/M-learning>
- Motiwalla, L. F. (2005). *Mobile learning: A framework and evaluation.* Noudettu osoitteesta <https://www.sciencedirect.com/science/article/pii/S0360131505001569>
- Mozilla. (1. 4 2019). *HTTP request methods.* Noudettu osoitteesta MDN web docs: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>
- Retta, G. (2009). *The Evolution of Mobile Teaching and Learning.*
- Roy, V. (16. 4 2018). *Quora.* Noudettu osoitteesta Quora: <https://www.quora.com/What-are-the-best-tutorials-to-learn-MERN-Stack>
- Sharma, P. (21. 06 2018). *Comprehensive Guide to build a Recommendation Engine from scratch (in Python).* Noudettu osoitteesta Analytics Vidhya: <https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-recommendation-engine-python/>
- Studies.* (2018). Noudettu osoitteesta Haaga-Helia: <http://www.haaga-helia.fi/en/exch/studies>
- Tegmark, M. (2019). *Benefits & Risks of Artificial Intelligence.* Noudettu osoitteesta future of life: <https://futureoflife.org/background/benefits-risks-of-artificial-intelligence/>
- Thinkwik. (6. 12 2017). *Why ReactJS is gaining so much popularity these days.* Noudettu osoitteesta Medium: <https://medium.com/@thinkwik/why-reactjs-is-gaining-so-much-popularity-these-days-c3aa686ec0b3>

U. Cortés, M. S.-M. (2000). *Artificial Intelligence and Environmental Decision Support Systems*.

*What is Artificial Intelligence*. (2018). Noudettu osoitteesta TechnoPedia:  
<https://www.techopedia.com/definition/190/artificial-intelligence-ai>

Wikipedia. (2019). *Data mining*. Noudettu osoitteesta Wikipedia:  
[https://en.wikipedia.org/wiki/Data\\_mining](https://en.wikipedia.org/wiki/Data_mining)

Wikipedia. (12. April 2019). *Machine Learning*. Noudettu osoitteesta Wikipedia:  
[https://en.wikipedia.org/wiki/Machine\\_learning](https://en.wikipedia.org/wiki/Machine_learning)

Wikipedia. (05 2019). *Machine Learning*. Noudettu osoitteesta Wikipedia:  
[https://en.wikipedia.org/wiki/Machine\\_learning](https://en.wikipedia.org/wiki/Machine_learning)

## 10 Appendices

### 10.1 Report structure

<b>Cover page, abstract, acknowledgments, table of contents, list of abbreviations</b>
<b>Introduction</b> <ul style="list-style-type: none"><li>– general introduction</li><li>– objectives</li><li>– presentation of functionalities and benefits</li></ul>
<b>Research questions and methodology</b> <ul style="list-style-type: none"><li>– theoretical questions</li><li>– problems and challenges</li><li>– development tasks and methodology</li><li>– test case with fabricated data</li><li>– study case with real users</li></ul>
<b>Related research</b> <ul style="list-style-type: none"><li>– original research paper</li><li>– different study phases of a student</li><li>– research about studying and teaching in a mobile context</li><li>– papers addressing the recommendation challenge using AI</li><li>– general researches about artificial intelligence</li></ul>
<b>Design</b> <ul style="list-style-type: none"><li>– mobile application design (mock-ups and prototype)</li><li>– website print screens (different components)</li></ul>
<b>Implementation</b> <ul style="list-style-type: none"><li>– project architecture (three-tier architecture)</li><li>– mobile application specificities (OS, language)</li><li>– website development (MVC architecture, languages)</li><li>– react.js components</li><li>– back-end server (API, node.js, structure)</li><li>– full stack application (MERN stack)</li><li>– components communication</li><li>– database choice (NoSQL) and solution (MongoDB)</li><li>– document-oriented stores</li><li>– collections and documents</li><li>– global architecture, user and components interaction</li><li>– student data</li><li>– courses, resources and quizzes statistics recommendation engine (artificial intelligence)</li><li>– data mining and machine learning</li></ul>
<b>Results</b> <ul style="list-style-type: none"><li>– study case results</li></ul>

<ul style="list-style-type: none"><li>- feedback, interview, ratings and suggestions</li><li>- test case results and evidence of efficiency</li><li>- cases conclusion and discussion</li></ul>
<b>Discussion</b> <ul style="list-style-type: none"><li>- answers to the research questions</li><li>- challenges of the project</li><li>- summary</li></ul>
<b>Recommendation</b> <ul style="list-style-type: none"><li>- artificial intelligence nowadays</li><li>- general feelings</li><li>- conclusions and suggestions for development or further work</li></ul>
<b>References</b>
<b>Appendices</b> <ul style="list-style-type: none"><li>- report structure</li><li>- study case report and analysis</li><li>- usability test plan</li><li>- test case results</li></ul>