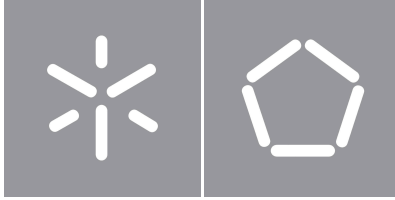


University of Minho
School of Engineering

João Pedro Neves Félix

**Modernising an e-learning
platform with a new
user interface**



University of Minho

School of Engineering

João Pedro Neves Félix

**Modernising an e-learning
platform with a new
user interface**

Master's Dissertation in
Informatics Engineering

Dissertation supervised by
João Miguel Lobo Fernandes

Copyright and Terms of Use for Third Party Work

This dissertation reports on academic work that can be used by third parties as long as the internationally accepted standards and good practices are respected concerning copyright and related rights.

This work can thereafter be used under the terms established in the license below.

Readers needing authorization conditions not provided for in the indicated licensing should contact the author through the RepositóriUM of the University of Minho.

License granted to users of this work:



CC BY-NC

<https://creativecommons.org/licenses/by-nc/4.0/> *[Esta licença permite que outros remissem, adaptem e criem a partir do seu trabalho para fins não comerciais, e embora os novos trabalhos tenham de lhe atribuir o devido crédito e não possam ser usados para fins comerciais, eles não têm de licenciar esses trabalhos derivados ao abrigo dos mesmos termos.]*

Acknowledgements

First and foremost, I would like to show my appreciation to my university. The invaluable guidance and support shared throughout my academic journey have shaped my knowledge and skills, preparing me for the challenges ahead.

I would like to express my deep gratitude to my supervisor, João Miguel Lobo Fernandes, whose guidance was invaluable. His profound insight and meticulous feedback have significantly shaped the trajectory of my research and writing.

To my professors, thank you for your devotion to teaching and mentorship. Your insights and knowledge have been instrumental in my studies. Your dedication to fostering academic and personal growth in your students is truly commendable.

I am deeply grateful to my colleagues at Altice Labs. Your understanding has made it possible for me to achieve my academic pursuits. The opportunity to apply my theoretical knowledge in a real-world setting, under your guidance, has been an enriching and invaluable experience.

To my family, thank you for your love and support. Your constant encouragement and patience have guided me throughout this journey.

I cannot express enough thanks to my friends for their encouragement throughout my studies. Your friendship, especially during the toughest times, has been a source of strength. I am truly fortunate to have you in my life.

In conclusion, this dissertation would not have been possible without you. Thank you for being part of this journey.

Statement of Integrity

I hereby declare having conducted this academic work with integrity.

I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

University of Minho, Braga, november 2023

João Pedro Neves Félix

Abstract

Distanced learning has a rich history and its evolution is closely connected to technology. It started with hand-written letters, later through radios, CD-ROMs, and now with the internet. With these advancements, the definition of e-learning appears - learning through the use of electronic devices. This new learning methodology comes with its own list of advantages (accessibility, lower cost, flexibility, etc.) and disadvantages (low motivation, social isolation, effectiveness, etc.). Larger institutions, such as universities and organisations, find e-learning very appealing mostly due to its long-term cost-efficiency.

Web accessibility and usability are important aspects of e-learning, rooted in the fact that the web is one of the most used means for sharing content. Ensuring that educational materials are accessible to all learners, regardless of their abilities is paramount to e-learning. Developers need to understand how can they make websites more accessible and usable, considering aspects such as design and implementation.

E-learning is now widely used, therefore a market appeared around it, paving the way for platforms such as Blackboard and Moodle. These platforms have carved their niche in the realm of digital education, each with its own strengths and unique features. By looking at what they offer we can deepen our understanding of what e-learning tools have to offer and what functionalities users value most.

Formare is an e-learning web platform created by Altice Labs. The company feels the platform is built using obsolete technologies (web forms) and wants to modernize the front-end layer of the application. Connecting Formare's functionalities to a new web interface (using React) is not a straightforward process, so an API that implements the back office and connects it to the front end was introduced.

This dissertation documents the implementation process of modernizing an e-learning platform with a new user interface, going in-depth about the problems and solutions found throughout development.

Keywords Distance Learning, E-learning, Web, Formare, Web Forms, React, API

Resumo

O ensino à distância possui uma rica história e a sua evolução está ligada à tecnologia. Começou com cartas, passou pelos rádios, CD-ROMs e agora a internet. Com estes avanços, surge a definição de e-learning - ensino através de dispositivos eletrônicos. Este novo tipo de aprendizagem traz consigo tanto vantagens (acessibilidade, custos mais baixos, flexibilidade, etc.) como desvantagens (baixa motivação, isolamento social, eficácia, etc.). Instituições, como universidades e empresas, encontram no e-learning uma metodologia de ensino atraente principalmente devido ao seu custo e eficiência a longo prazo.

A acessibilidade e usabilidade na web são aspetos importantes do e-learning, devido ao facto de a web ser um dos meios mais utilizados para partilhar conteúdos. Garantir que os materiais educativos são acessíveis a todos os alunos, independentemente das suas capacidades, é fundamental para o e-learning. Os desenvolvedores precisam de compreender como podem desenvolver websites mais acessíveis, considerando aspetos como o design e a implementação.

O e-learning é agora amplamente utilizado, o que deu origem a um novo mercado, abrindo caminho para plataformas como Blackboard e Moodle. Estas plataformas criaram o seu espaço no campo da educação digital, cada uma com as suas próprias vantagens e características. Ao analisarmos o que elas oferecem, podemos aprofundar a nossa compreensão das ferramentas e-learning e das funcionalidades mais valorizadas pelos utilizadores.

O Formare é uma plataforma de e-learning criada pela Altice Labs. A empresa considera que a plataforma foi construída utilizando tecnologias obsoletas (web forms) e pretende modernizar a camada front-end da aplicação. A conexão das funcionalidades do Formare a uma nova interface web (usando React) não é um processo simples, por isso foi introduzida uma API que implementa o back office e a conecta à camada front-end.

Esta dissertação documenta o processo de implementação da modernização de uma plataforma de e-learning com uma nova interface, aprofundando os problemas e soluções encontrados ao longo do desenvolvimento.

Palavras-chave Ensino à distância, E-learning, Web, Formare, Web Forms, React, API

Contents

1	Introduction	1
1.1	Context	1
1.2	Motivation	4
1.3	Objectives	4
2	State of the Art	6
2.1	E-learning in recent years	6
2.1.1	E-learning and network technologies	7
2.1.2	Advantages and disadvantages	9
2.1.3	E-learning in organisations and universities	11
2.2	Accessibility and usability	14
2.2.1	Accessibility	15
2.2.2	Usability	17
2.3	Other e-learning platforms	19
2.3.1	Moodle	19
2.3.2	Blackboard	20
3	Architecture and Requirements	22
3.1	Project technical description	22
3.2	Architecture	23
3.3	Requirements	25
3.3.1	Architecture software requirements	26
3.3.2	Authentication software requirements	26
3.3.3	Account creation software requirements	27
3.3.4	Password recovery software requirements	27
3.3.5	Support for multi-language software requirements	28

3.3.6	User profile related requirements	28
3.3.7	List of courses software requirements	29
3.3.8	List of notices software requirements	29
3.3.9	List of contents software requirements	29
3.3.10	List of questionnaires software requirements	30
3.3.11	Questionnaire instructions software requirements	30
3.3.12	Answering and submitting questionnaires software requirements	31
4	Use Cases and Implementation	33
4.1	Integration	33
4.1.1	Configuration files	33
4.1.2	Injecting the services	33
4.1.3	Default implementations	34
4.1.4	Requests and responses	34
4.1.5	Distinguishing between client and server logic	35
4.2	Use cases	36
4.2.1	Authenticate	36
4.2.2	Account creation	38
4.2.3	Logout	39
4.2.4	Password recovery	40
4.2.5	Profile	41
4.2.6	Multi-language	41
4.2.7	List of courses	43
4.2.8	Course menu	44
4.2.9	List of notices	44
4.2.10	List of contents	46
4.2.11	Display content of type link	47
4.2.12	Display content of type file	47
4.2.13	Display content of type non-standardised	48
4.2.14	List of questionnaires	49
4.2.15	Display questionnaire	49
4.2.16	Save questionnaire answers	51
4.2.17	Submit questionnaire answers	52

4.2.18	Abandon questionnaire	54
5	Final Results	55
5.1	Authentication and profile	55
5.2	Courses	56
5.3	Notices	57
5.4	Content	58
5.5	Questionnaires	59
5.6	Architecture and Multi-language support	61
6	Future Works and Conclusion	62
6.1	Future work	62
6.1.1	Missing functionality	62
6.1.2	Design	63
6.1.3	Testing	63
6.2	Conclusion	64

List of Figures

1	Evolution of the network technologies Kahiigi et al. [2008]	8
2	Evolution of e-learning technologies Kahiigi et al. [2008]	8
3	Evolution of content development Kahiigi et al. [2008]	9
4	Functional Suitability Ouadoud et al. [2016]	20
5	Previous architecture	24
6	Current architecture	25
7	Authentication and profile pages	56
8	Courses page	57
9	Content related pages	59
10	Questionnaire related pages	60

List of Tables

1	Architecture software requirements	26
2	Authentication software requirements	27
3	Account creation software requirements	27
4	Password recovery software requirements	28
5	Support for multi-language software requirements	28
6	User profile software requirements	29
7	List of courses software requirements	29
8	List of notices software requirements	29
9	List of contents software requirements	30
10	List of questionnaires software requirements	30
11	Questionnaire instructions software requirements	31
12	Answering and submitting questionnaires software requirements	32
13	Authenticate use case	37
14	Account creation use case	38
15	Logout use case	40
16	Password recovery use case	40
17	Profile use case	41
18	Multi-language use case	42
19	List of courses use case	43
20	Course menu use case	44
21	List of notices use case	45
22	List of contents use case	46
23	Display content of type link use case	47
24	Display content of type file use case	48

25	Display content of type non-standardised use case	48
26	List of questionnaires use case	49
27	Display questionnaire use case	50
28	Save questionnaire answers use case	52
29	Submit questionnaire answers use case	52
30	Abandon questionnaire use case	54
31	Unfulfilled contents software requirements	59
32	Unfulfilled questionnaires software requirements	61

Chapter 1

Introduction

The current document details the context, motivation, objectives, state-of-the-art analysis and implementation for the new front-end of an e-learning platform (Formare). This project exists as a Master's Dissertation in Informatics Engineering at Universidade do Minho and was proposed by Altice Labs. In this first chapter, is a contextualisation of this dissertation's project and the motivation behind it, as well as the objectives to achieve.

1.1 Context

With education being vital in the knowledge society, new ideas in the area of learning keep appearing. [Kahiigi et al. \[2008\]](#) describe Learning as a process through which learners achieve their learning goals by carrying out several learning activities and participating in interactions to reflect their understanding. Thus, learning seems to result from a change in students' perception of reality related to the problem area under study.

The authors also outline that students learn differently and how information is presented to them affects their ability to learn. Three different learning styles are then presented:

- **Visual Learners** learn best through seeing things such as images, demonstrations, facial expressions, and the body language of the instructor.
- **Auditory Learners** learn best by hearing things through verbal lectures and discussions.
- **Tactile Learners** learn best through experiencing, reflecting, interacting, and doing things.

Not only are there three different learning styles, but also different learning methods [[Kahiigi et al., 2008](#)]:

- **Traditional Learning** refers to face-to-face sessions, through which the teacher delivers course material to students in the same place and at the same time. The learning method is considered teacher-centred, where the teacher focuses on providing the learning information to the students.
- **Distance Learning** is a form of education where the main elements include the physical separation of teachers and students during instruction and the use of various means to facilitate student-teacher and student-student communication [[Berg and Simonson, 2016](#)].
- **Blended Learning** uses a combination of various learning methods that include face-to-face classroom activities, live e-learning and self-paced learning.
- **Mobile Learning** is defined as the learning or delivery of content that is facilitated by the use of portable technologies such as mobile phones.
- **Personalized Learning** is a learning approach that facilitates and supports individualised learning. Each learner has a learning path that caters for the learner's learning needs and interests in a productive and meaningful way.

Due to the nature of the dissertation's problem, around e-learning, it becomes essential to take a deeper look at Distance Learning.

As said before, Distance Learning is a learning methodology focused on complementing the distance between student-teacher and student-student. It is easy to think that this learning can only exist with the help of computers; however, Distance Learning has been around for far longer.

Its first known landmark in history was in 1728 by Caleb Phillips, a shorthand teacher who posted an announcement in the Boston Gazette inviting students to learn his techniques through letters [[Tait, 2018](#)]. The first correspondence school focused on teaching languages was founded in 1856 in Berlin. Soon after, other universities started incorporating full correspondence courses (such as Wisconsin University in 1891). Due to social demand, distance Learning became even more prevalent after the First World War [[Suely Trevisan Araújo, 2010](#)].

The evolution of communication services and the invention of the radio significantly impacted Distance Learning. Instead of teaching through letters, the radio was the go-to medium and it had great success, particularly in Latin America.

With the rapid evolution of technology, by the '60s and '70s, Distance Learning had gone a long way since using just the radio. At this point, the radio was no longer the only medium being used. Television, videotext, VCRs, radio, etc. were all used to allow for Distance Learning and played a big role in classrooms [[Sadeghi, 2019](#)].

Technology has played such a crucial role in Distance Learning that today, it is hard to imagine one without the other. Having said that, I can now talk about e-learning.

As noted by [Guragain \[2016\]](#), e-learning is an inclusive term which generally covers a wide range of teaching and learning activities via electronic media such as computers or web-based platforms, however, it is not necessarily distance learning and may also occur in face-to-face classes.

The term usually refers to using different kinds of information and communication technologies in education. Teachers and students are typically separated from one another and use technology to communicate and to help their learning activities [[Tayebinik and Puteh, 2013](#)].

E-learning has been a particularly important tool in organisations, as in most of them technology is the driving force of workplace training. This is to be expected since e-learning as a means of training benefits the organisations that employ it [[Derouin et al., 2005](#)]. These benefits are further discussed in the next section.

Universities have also been making use of e-learning tools before the COVID-19 pandemic. Competitiveness among universities for students, the fact that many students are now working during their studies and the increasing demand for degrees to make a living are all facts that contribute to the growth of fully online courses. Virtual universities have also become quite popular over the years [[?](#)]. Blended methods, however, are the norm since most classes are primarily taught in a traditional method (face-to-face) and then make use of e-learning tools to share studying materials between students.

Primary, middle and secondary schools, on the other hand, have mainly used traditional education (face-to-face education) before COVID, thus largely ignoring e-learning tools. The pandemic, however, forced these institutions to evolve and adapt.

The pandemic was declared by the [World Health Organization \(WHO\)](#) on 11 March 2020, which drove the population to confinement, therefore, affecting many sectors, one of them being education. The Organisation for European Economic Co-operation (OECD) then released a report proposing that new plans be made to allow students to continue their education. As said in [Lynch \[2020\]](#), this drove institutions to change from traditional education to distance education through online technologies.

This chain of events has put e-learning at the forefront of education during the pandemic and played a big part in mitigating its negative impact on education. Both students and teachers have now found new tools that can help teach and learn without needing a classroom [[Cordeiro, 2020](#)].

1.2 Motivation

As e-learning grows in popularity, so does the need for platforms that facilitate its tools. Any organisation and/or institution that intends to implement e-learning as an efficient means of education and training will lean toward an e-learning platform – Universidade do Minho, as an example, uses the Blackboard platform.

Altice Labs, a company belonging to the Altice group, was created in 1950 as the Study Group for Automatic Switching when an engineering team was created to internally develop the most advanced telecommunications technology of its time and has an e-learning platform of its own, Formare [[Labs, 2023](#)].

As described on the official website [Formare \[2020\]](#), "Formare is a Learning Management System which supports training and education solutions in e-learning and b-learning (combination of face-to-face and online education) environments. It is adaptable to different contexts and training methodologies and follows technological innovation at the service of education and training, like the use of Mobile Learning and personalized learning environments. All features provided by Formare are the result of users' requests and are organized in modules, enabling an intuitive and effective interaction with all the system's users".

This platform can be used on both desktop and mobile platforms; however, functionalities differ between the two – the desktop version is the most complete and includes all the management features, while the mobile one focuses only on students.

Some clients have shown concerns regarding the application's current state. Its front end is outdated both from a design and technical standpoint and leaves a lot to be desired when it comes to accessibility and usability. Therefore, Altice Labs is interested in having someone who would rework Formare's front end to be per its client's wishes.

That being said, modernising both the desktop and final versions requires more time than this thesis' final date allowed; thus, only the mobile version was worked on. By the end of this project, it was possible to have a new mobile front end, that follows common accessibility and usability guidelines and to have an API that allows for such an implementation.

1.3 Objectives

Given the motivation behind this project, one main object can be highlighted for this thesis:

- **Modernise Formare's user interface**

To accomplish this goal there must be some time put into research. Topics include the recent evolution of e-learning and its connection to technology, the advantages and disadvantages of using e-learning technologies and methodologies and its use in both universities and organisations. Accessibility and usability are also important subjects to be researched, as well as other e-learning platforms, more specifically, Moodle and Blackboard Learn.

Implementation-wise, a new front-end layer must be built with a new, more recent web framework (React) as well as a new styles library, made by Altice Labs. This new layer must be integrated with the existing application logic. A reliable way to make Web Forms and React work together was not found; therefore, a new .NET API layer was also built to mediate between the user interface and the back office.

Chapter 2

State of the Art

This chapter begins by discussing the evolution of e-learning in recent years and what impact advancements in technology have had on it.

I also look into the advantages and disadvantages we can expect from the use of this learning method, as well as how we can increase its efficiency.

Eventually, I discuss how e-learning is being used today in both universities and organisations and what results these institutions have been having with its implementation.

Then I examine what are the best practices regarding usability and accessibility and how can one make their websites more accessible and usable.

Finally, I look at the current market of e-learning platforms, both proprietary and open-source, paying special attention to their use cases and functionalities.

2.1 E-learning in recent years

I said before that Distance Learning has been a valid education method for almost 300 years, but due to its close history with technology, this concept has been closely entangled with e-learning since the latter naturally suits the former [[Al-Arimi, 2014](#)].

With that said, many factors, such as the subject matter, the knowledge of the audience and the environment around the audience, have affected e-learning growth from the early times to the present.

In the early '80s, computers were not widely available nor did many people know how to use them. Because of this, instructor-led training was the only way students could learn from or about computers. Institutions owned most computers and could only be used during office hours for official reasons; therefore, if staff wanted to learn how to use them they needed to use their work hours which led to slowdowns in their official work, being counterproductive to offices [[Guragain, 2016](#)].

When the era of multimedia advancements began, from around 1985 to 1995, and technologies

like CD-ROMs and PowerPoint started to become popular, instructors began using these technologies to make training portable and visually engaging. This resulted in a major upgrade from instructor-led training, leading to cost savings and being more productive since CD-ROMs could be easily transported and repeatedly delivered without the presence of an instructor [Gunawardena and Mclsaac, 2013].

When the **World Wide Web (WWW)** started evolving, during the mid-'90s, the possibility of using this new technology to deliver information in more efficient and cost-effective ways began to be considered. Thus, during this period, email, web browsers, HTML, media players, etc. saw a big growth in popularity among both students and teachers. There were, however, still some things that needed to be accomplished with these new technologies due to the large size files can have [Guragain, 2016].

Then in the early 20th century, there was major growth in the field of computers and the web, which inevitably changed the way people learn. Student-teacher and student-student communication could be easily done in real-time via the web and with the improving accessibility of these technologies, information was easily acquired. [Bari et al., 2018].

2.1.1 E-learning and network technologies

It has been established that e-learning evolution comes mainly due to technological evolution.

Kahiigi et al. [2008] says that network technologies are especially important since they facilitate content presentation, personalization and ubiquitous learning. Their study makes clear the relationship between these two technologies.

Client-server architecture has allowed learners to access study materials from a centralised server. The further development of this architecture has resulted in the inclusion of web-based features leading to the emergence of Internet/Intranet/Extranet technologies that support these features. Developments in these technologies have now permitted the use of portable devices in the learning process through wireless broadband technologies. Figure 2 shows these developments over the years:

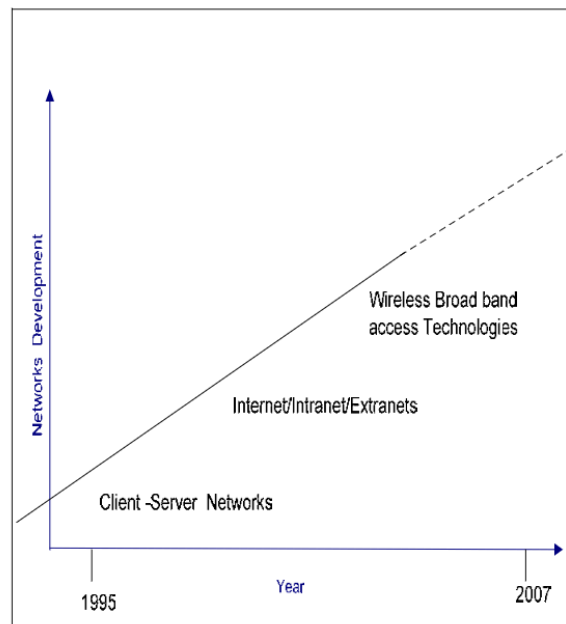


Figure 1: Evolution of the network technologies [Kahiigi et al. \[2008\]](#)

From CD-ROMs to personalised technologies, e-learning has seen a huge evolution in the past decade. CD-ROMs provide portable content that is not facilitated by network technologies and therefore were popular mostly in a time when these technologies were not yet easily accessible. Later, when network technologies were more developed, other e-learning tools started to gain traction such as LMS/CMS/LCMS (learning management system, content management system, learning content management system), multimedia and virtual communities, game authoring tools and personalised tools. Figure 3 shows this development over the years:

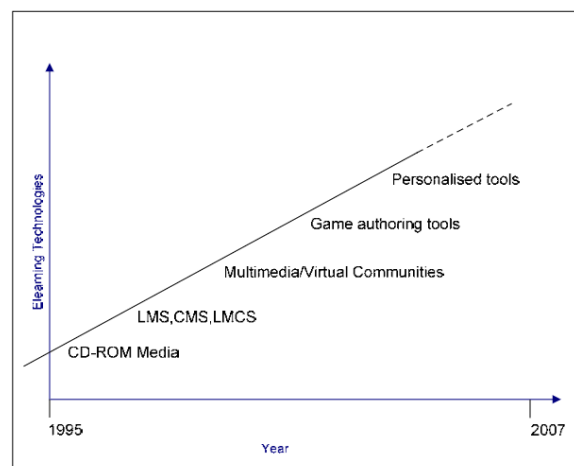


Figure 2: Evolution of e-learning technologies [Kahiigi et al. \[2008\]](#)

The evolution of e-learning technologies has also encouraged the evolution of the content accessible by these technologies, as illustrated in Figure 4.

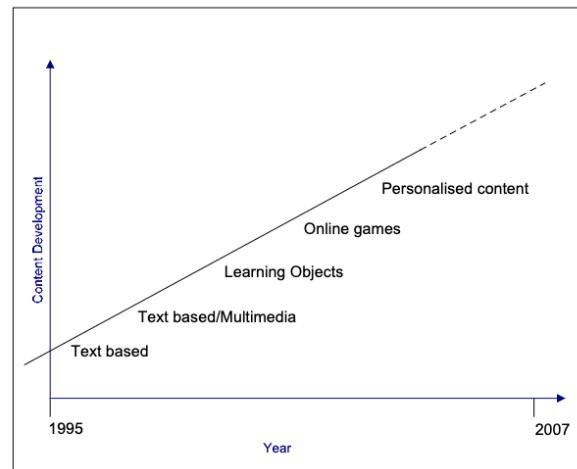


Figure 3: Evolution of content development [Kahiigi et al. \[2008\]](#)

This study made by [Kahiigi et al. \[2008\]](#) makes evident how network technologies allowed for the evolution of e-learning tools and the content these tools provide.

2.1.2 Advantages and disadvantages

As has been established in the previous section, e-learning has evolved very rapidly in recent times due to its deep connection to technology development. As the world is moving towards the digital age, more institutions are trying to maximise the use of technology in education, therefore both the advantages and disadvantages of these tools must be considered carefully. [Guragain \[2016\]](#) lists the following advantages and disadvantages.

Advantages

E-learning has many advantages. Combined with a well-organised e-learning system and a highly motivated student, one can achieve great success in a short time. Some of the major advantages of e-learning, by [Arkorful et al. \[2015\]](#), [Al Rawashdeh et al. \[2021\]](#) and [Radović-Marković et al. \[2010\]](#), are:

- **Easily Accessible** - e-learning materials can be accessed at any time the learner wants since the learner is not required to be present in a classroom.
- **Lower cost** - e-learning is cost-efficient for most students, as they can choose from an extensive range of courses and select those that cater to their needs. Many universities can also take

advantage of e-learning cost-efficiency because learning platforms and materials can be reused.

- **Up-to-date learning materials** - study materials in e-learning systems, once uploaded, can be reused for longer periods than those in classroom-based education systems. They can also be updated more easily and more frequently making them easier to maintain.
- **Flexible way of learning** - most of the e-learning study materials are stored for the students to access whenever they want, at a pace they think is preferable, allowing them to, for example, skip over materials they already know. Students can also choose between an instructor-led and a self-learning system.
- **World-wide learning society** - because e-learning materials can be accessed anywhere, students and teachers from different geographical locations can help each other or even contribute to the studying materials themselves to keep them updated. Therefore, it can be said that e-learning systems help in creating a worldwide learning society.
- **Scalable e-learning systems** - the number of students in e-learning systems does not significantly affect its total cost.
- **Higher degree of freedom for students** - e-learning systems provide the possibility for students to learn the same material repeatedly until they are satisfied.
- **Better retention** - the video and audio materials used in e-learning make the whole learning process more fun. This will help students to remember the things they learn for a long period. Retention is also made easier because e-learning materials can be accessed at any time, allowing for repetition.

Disadvantages

As with any other system, e-learning also has some drawbacks. Some of the major disadvantages of e-learning, by [Al Rawashdeh et al. \[2021\]](#), [Radović-Marković et al. \[2010\]](#) and [Akhter et al. \[2021\]](#), are:

- **Low motivation** - students are responsible for the routine and organisation of the course, which can lead to low motivation. Students with low motivation may only achieve the set goals sometimes.
- **Technology-dependency** - e-learning studying materials are delivered through computer applications, therefore it is necessary to know how to use these technologies to successfully use e-learning

tools. Other factors such as a poor Internet connection and machine malfunction may negatively impact the learning experience.

- **Compatibility issues** - the study materials prepared using one system may not be compatible with another and not all websites are free to access. The geographical location could also affect accessibility as some materials may be accessible only in certain countries.
- **Reliability of the content** - this one happens mostly when considering e-learning as every action that leads to learning using technology (browsing the web). Not all content available on the internet is reliable and readers must be careful.
- **Social isolation** - since e-learning is made mostly via distanced education, students may feel socially isolated due to the lack of interactions with other people while learning.
- **Expense management** - in the long run, e-learning is usually a cheaper option, however, it is a big investment for institutions that do not already have the necessary equipment, such as computers, projectors and software. Also, the cost of developing training materials is high compared to traditional methods.
- **Disadvantages for disabled students** - e-learning studying materials are made to cater to the needs of most students to allow for high reusability and low development costs. This, unfortunately, could put disabled students, such as the visually impaired, at a disadvantage, since the costs of developing adapted content may be high.
- **Not effective in all cases** - there may be some cases where e-learning is not very effective, such as students who prefer face-to-face education or studying materials that are more adequate when combined with two-way communication.

2.1.3 E-learning in organisations and universities

E-learning as a teaching and training methodology with long-term cost-efficiency makes it very appealing for larger institutions with a lot of employees/students such as organisations and universities.

Organisations

In organisations, e-learning is a very useful tool for workforce training. As said by [Derouin et al. \[2005\]](#), it allows them to deliver training consistently to all employees, update training content when necessary,

reduce travel costs to outside training facilities and provide training to employees on demand, anytime, anywhere. They go on to mention that 95% of respondents to a 2003 survey by the [American Society for Training and Development \(ASTD\)](#) reported using some form of e-learning in their companies.

This is of course not surprising since e-learning is a very good way of delivering instructions using varied technologies, such as live-streaming audio and video technologies, textual materials in the form of electronic PowerPoint slides, and discussions through the use of message boards and chat rooms. Not only are these organisations using e-learning as a means of training the IT skills of their employees, but also using them to teach their collaborators both soft and business skills such as management, leadership, communication, customer service, quality management and human resources skills [[Sambrook, 2003](#)].

Another appealing fact about e-learning as a means of training employees is its self-paced nature, allowing trainees to study materials as fast or slow as they want [[Zahari et al., 2020](#)]. This seems to have a positive, but small, impact on learning outcomes. This, however, comes with some drawbacks. Since trainees can choose how to tackle studying materials, they have to constantly make choices about what order they should study materials in and what materials are more important than others [[Schreurs and Al-Huneidi, 2012](#)]. This obviously can lead students to take a less-than-optimal approach to their learning experience, making it important for organisations to ensure that the amount of control trainees have is consistent with the goals of the instructional program.

One other thing that very much goes into the needs of organisations is how rapidly and easily e-learning content can be updated. Training materials can become outdated rather quickly and must be, more often than not, developed in short time frames, making e-learning a perfect option for organisations since it covers those needs. [Harun \[2001\]](#) describe how their Continuing Medical Education web-based portal provides up-to-date learning resources.

Recently, organisations have been looking for ways to keep trainees engaged since this is one of the biggest problems fully distanced e-learning courses face. One way this problem has been tackled is through the use of games as a means of training and the gamification of courses, adding features such as scoreboards. The use of stories or narratives in e-learning can also promote learner engagement [[Derouin et al., 2005](#)].

The fact is that e-learning technologies have become inescapable to organisations that need to employ any kind of training. Even if they did not go with the full online e-learning approach, they are most likely using a more blended methodology for their training sessions. I have experienced two very different approaches to training. Altice Labs, for example, has opted for a mostly distanced, asynchronous approach with a few synchronous sessions where collaborators can partake face-to-face or at home. Critical Software, on the

other hand, prefers a mostly synchronous, blended approach where some sessions require face-to-face participation and others allow the collaborators to watch them at home.

Universities

Just as it was for organisations, e-learning is a tool that brings great advantages to a higher education context. It gives opportunities to students who lack them due to factors such as money, distance and time.

[Kahiigi et al. \[2008\]](#) attributes the transition of universities from a more traditional focused learning methodology to a more recent approach, that also uses e-learning, to:

- The rising of a knowledge society and knowledge-based economy, and the drive to join them.
- The opportunities presented by e-learning to meet student needs at a reduced cost.
- The growing demand for personnel in the labour market.
- The increasingly higher number of students on-campus, off-campus, life-long and on-the-move who want to continue their education despite their adversities.
- The ease of availability of electronic learning resources.
- The higher demand for alternative learning methods.
- Collaborative research opportunities.

This change in methodology has caused a paradigm shift from teacher-centredness to student-centredness. In this case, the teacher takes on a facilitator role while the students take ownership of their learning and personal development.

Despite the great benefits that e-learning can bring to these institutions, their adoption is not always easy; there may be difficulties in implementing these systems, such as reluctant personnel due to changes in work patterns or inexperience with technology, implementation problems due to cost or infrastructure, and many others [[Kibuku et al., 2020](#)].

[Kahiigi et al. \[2008\]](#) also says that, even in universities that have successfully implemented e-learning tools, there have been accounts of emerging issues such as the following:

- Despite e-learning facilitating learning for students, it does not mean there is an improvement in learning quality.

- Both teachers and students are required to have a sufficient understanding of information and communication technologies for e-learning tools to be practical.
- Traditional learning has been around for far longer than e-learning and is the standard. That may create doubt in teachers' minds about e-learning's usefulness, ultimately resulting in a reluctance to adopt new methodologies.
- Successful e-learning implementations require a high level of commitment, support and maintenance from institutions and that may not always be given to these tools.

Despite these drawbacks, the future of higher education is undoubtedly alongside e-learning. Most universities are already using some of these tools, either on a blended approach [Aguti et al., 2014], fully online, or both. This shift has been sped up by the recent pandemic, which forced many universities to completely shut down all classroom-based learning methodologies in favour of distance learning through e-learning tools. Universidade do Minho, from my personal experience, has always used e-learning tools to share studying materials with students while maintaining a mostly traditional learning approach. However, the university completely shifted from a mostly traditional approach to a fully online one through e-learning during the pandemic and consequent lockdown. At present, because teachers have seen how useful these tools can be, some have started to regularly use them, not only to share materials but also for classes, exercises, tests, etc.

2.2 Accessibility and usability

With the rapid evolution of networking technologies, the web has taken over the world in recent years. Matter of fact, most people nowadays surf the web and its contents daily, making it a vital medium to be in for corporations due to the sheer amount of users that companies can engage with.

That said, it is not enough for companies to just put their services on the web; those services must be as easy and comfortable to use as possible, but also accommodate any type of user that may come to use them. Thus, accessibility and usability become unavoidable terms when it comes to web products.

Note that this section is only meant to show that I am aware of what accessibility and usability are, their importance, and how they can be achieved.

2.2.1 Accessibility

'Web accessibility means that websites, tools, and technologies are designed and developed so that people with disabilities can use them. It also benefits people without disabilities' [[WAI, 2022](#)].

According to [Statista \[2022\]](#), in 2021 the percentage of people worldwide with access to the web is 63%. With such a majority of people connected to the internet and its contents, it's no news that this resource has become very important in many aspects of life such as education, employment, commerce, health and others. Such importance means that accessibility is an important matter when it comes to equality in opportunities.

There is also a strong business case for accessibility, since making the web more accessible to users with disabilities often ends up improving user experience and satisfaction [[Paciello, 2000](#)], especially when it comes to older users. Therefore, it may improve branding, innovation and market reach.

Components of web accessibility

In a correct implementation of web accessibility, there are many components to consider. [WAI \[2022\]](#) lists the following:

- Developers are the ones making the websites.
- Users are the ones using the website.
- Content is the information displayed on websites.
- Evaluation tools - web accessibility evaluation tools.
- Authoring tools - software that creates websites.
- Assistive technology - screen readers, alternative keyboards.

These components interact with each other in the following way: web developers usually use authoring tools and evaluation tools to create web content. Users, on the other hand, through assistive technologies, interact with the content.

It is important to note that each component plays an important part in making a website accessible. If one is missing a feature, there is little to no incentive to implement it on the other components, which ends up resulting in a lesser experience for users and poor accessibility or even inaccessibility. Inaccessibility, especially, is something to be wary of because it means some users with certain disabilities cannot use a particular website.

Accessibility principles

We have already established why accessibility is an important term to keep in mind and what components go into making an accessible website; however, we have yet to cover the accessibility principles that guide web developers. These are some of the principles outlined by [WAI \[2022\]](#):

- **Perceivable information and user interface**

- Non-text content should have text alternatives that convey the purpose of the content.
- For multimedia, content captions and other alternatives provide a basic level of accessibility.
- Content should be allowed to be presented in different ways, them being, read aloud, enlarged, or even adapted to meet the needs of different users.

- **Operable user interface and navigation**

- Some users do not rely on the mouse, using mainly the keyboard to navigate web pages, therefore all functionalities should be available through the keyboard.
- Some people need more time than others to read and use the content, thus an effort should be made so that there is enough time to read and use the content displayed.
- Content should be well organised so that it can be easily navigated and understood by the users.

- **Understandable information and user interface**

- Ensuring that text content is easily readable and understandable and that it can be read aloud by text-to-speech.
- Some users may find themselves easily distracted by inconsistent appearance or behaviour, therefore content should appear and operate in predictable ways.
- Users may find forms and other interactions can be confusing or difficult to use and as such may be more likely to make mistakes. It is important to help users to avoid or understand such mistakes.

- **Robust content and reliable interpretation**

- Content should be compatible with current and future tools.

In sum, the internet is used by many different people, some need more help than others to interact with the content on the web. It is important to keep accessibility in mind when creating web pages to make sure that everyone has the same opportunities when it comes to the web.

2.2.2 Usability

When it comes to creating web content, usability and accessibility are terms that come hand in hand. As outlined in [WAI \[2022\]](#), 'their goals, approaches, and guidelines overlap significantly'.

While accessibility addresses the needs of people with disabilities, usability is about making navigating through that same content as effective, efficient and satisfying as it possibly can. Usability is about improving the user experience of every user, therefore it is insufficient to address the needs of disabled users. As [Abushamleh and Jusoh \[2021\]](#) mentions, usability is defined in ISO 9241:11 standards as 'The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in specific contexts of use'.

Because these two terms are so close to each other, it is often easier for developers to address them together when building web content. Most times, when developing a web page, decisions made to improve accessibility constantly improve usability as well.

Usability heuristics

Jacob Nielsen is the current principal of the Nielsen Norman Group, a UX research and consulting firm trusted by leading organisations worldwide to provide reliable guidance on user experience, and he has outlined ten usability heuristics for user interface design [[Nielsen, 2020](#)].

- **Visibility of system status**

- It is important to always show the user what is going on with the system. If the users know what happens after they take a certain action, it is easier for them to understand how the system works.

- **Match between the system and the real world**

- Design should depend on what types of users a web page is intended for. It is important to use terms and concepts that users are familiar with. Images and icons should follow the same rules. This should result in an intuitive experience.

- **User control and freedom**

- Mistakes are normal and as such, it is important to allow users to easily go back when an unwanted action is made. This protects users from getting stuck and feeling frustrated.
- **Consistency and standards**
 - Users use an array of different products. If a developer's website follows common standards, new users can efficiently start using their products right away.
- **Error prevention**
 - Error messages are important, however, [Nielsen \[2020\]](#) points out that the best designs are the ones that avoid problems in the first place.
- **Recognition rather than recall**
 - Elements, actions and options should be visible. Users should not need to remember information from one interface to another.
- **Flexibility and efficiency of use**
 - Experienced users should be given the option to speed up their interactions. One way of doing this is creating shortcuts that these types of users can use.
- **Aesthetic and minimalist design**
 - There should be an active attempt by designers to make sure their websites are not cluttered by useless information. The visual elements should support the users' goals.
- **Help users recognize, diagnose, and recover from errors**
 - I have said before that error messages are important, but better if avoided. When they, however, cannot be avoided they should be easy to understand and plainly explain the problem and possible solutions.
- **Help and documentation**
 - Some systems are too complex to be used efficiently without any additional help. Thus, it may be good for some systems to provide easy-to-read and concise documentation that focuses on user needs.

Jacob Nielsen does say that these are not specific usability guidelines, however, they are still worth taking into consideration.

2.3 Other e-learning platforms

This thesis' problem focuses on the implementation of a new front end for the e-learning platform Formare. Therefore, some time was dedicated to looking at other applications that serve the same purposes. In this section, I discuss several aspects of the open-source e-learning platform Moodle and retell my experience using the Blackboard e-learning platform as a student enrolled in Universidade do Minho.

2.3.1 Moodle

Moodle, developed by Dr. Martin Dougiamas, is an open-source learning management system (LMS) designed to support educators in creating and managing online courses. It is the preferred platform for IT teaching in elementary and secondary schools due to being free of charge and open-source.

In this platform, functionalities are separated into modules, such as site management, user management, course management, task modules, chat room modules, selecting modules, forum modules, logging modules, test modules, resource modules, and more. These modules can be integrated and applied in course design.

Among the many technologies used to build this platform, I want to highlight the following [\[Jin, 2012\]](#):

- Apache is a stable and open-source HTTP server. It can run on most computer operating systems. Moodle platform uses Apache because it is free of cost.
- PHP is a highly efficient web server which can run on all mainstream operating systems and web servers. Moodle uses this technology because, along with MySQL, it is compatible with various operating systems.
- MySQL is a fast and stable database. MySQL is the default database due to its compatibility, performance and nice user experience combined with PHP and Apache.

Concerning its functionalities, [Ouadoud et al. \[2016\]](#) compares four free online e-learning platforms, one of them being Moodle. I wish to specifically bring attention to the functional suitability section, which shows 'the degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions'. The next image shows what functionalities are provided, according to this study.

Functional suitability				
Platform's name:	Atutor*	Claroline Connect*	Moodle*	Sakai*
Functional completeness				
Forum	Yes, discussion forum / export [17]	Yes, public or private discussion forum [19]	Yes, Different types of forums are available and open to all [15]	Yes [7]
Synchronous causerie (chat)	Yes, flexible synchronous interaction in writing, audio and video [17, 18]	Yes	Yes, flexible synchronous interaction in writing [15]	Yes, [5]
Virtual Classroom (webinar)	Yes, 'BigBlueButton' plugin [18]	No	Yes, 'BigBlueButton' plugin	Yes, 'BigBlueButton' plugin [8]
Sharing documents	Yes, 'Dropbox' tool for creating shared files	Yes, collaboration Wiki space [19], and shared files.	Yes, collaboration Wiki space [15]	Yes, collaboration Wiki space [7], + 'Dropbox' tool [5]
Calendar	Yes, structure an agenda with tasks and deadlines [17]	Yes, structure an agenda with tasks and deadlines	Yes	Yes [7]
Awareness	Yes	Yes	Yes	Yes
Tests management	Yes, module 'Tests and Surveys'	Yes, generation of exercises (quizzes, MCQ, open answers ...) [19]	Yes, choose the level of sharing issues [15]	Yes, the tool open and closed questions. [6]
Collaboration (Wiki)	Yes, plugin 'EWiki' + implemented of ErFurtWiki [18]	Yes [19]	Yes, 'Wiki' collaboration space [15]	Yes [7]
Learners' management (registration, schedule...)	Yes, multiple authentication mechanisms of registered and unregistered learners [17]	Yes, user management space + interfacing with LDAP directory [19]	Yes, several authentication mechanisms through the module 'User Management' [15]	Partial, a space of role management / lack of planning
Learners management in working groups	Yes, Manual and automatic groups [17]	Partial, creation of related discussion forums and not groups	Yes, attribution of learners in groups (class)	Yes, creation of a discussion forum in groups
Users roles management	Yes, multi-profile (Student ; instructor ; administrator)	Yes, roles management space is offered by the platform [19]	Yes, multi-profile (Manager ; course creator ; Teacher ; Student ; administrator)	Yes, multi-profile management
Customizable platform	Yes, management roles, groups, users, modules, etc.	Yes, management roles, users, groups, plugins... [19].	Yes, roles management, users, groups, customizable themes [13], + installable activity plugins [15]	Yes
Advancement Scale or progression percentage in the course	Yes, module 'statistics'	Yes, consult the statistics of attendance and success in exercises	Yes, Registration and follow-up of activities [15]	Yes [6]
Resource Management (course)	Partial, sharing courses for the groups [17] and not in the activity spaces	Yes, share resources of activity spaces [19]	Yes, course management plugin that allows deposition of courses [15]	Yes, integration of multimedia in the course [5]
Support of multiple authors	Yes, the teacher has the right to create and manage its course	Yes, the teacher has the right to create and manage its course	Yes, the teacher has the right to create, manage, and give the access right to tutors in its course	Yes, the teacher has the right to create and manage its course [14]
Functional correctness				
Management of working time of learners and teachers	No, the platform does not support the working time of the learners	No, the platform does not support the working time of the learners	No, the platform does not support the working time of the learners	No, the platform does not support the working time of the learners
Results and notes	Yes, module 'Tests and Surveys'	Yes	Yes, All given notes in forums, tests and homework [15]	Yes [14]
Notes display	Yes	Yes	Yes	Yes
Tracking statistics of course	Yes, 'tests and surveys' tool	Yes, tools of monitoring the learners' activities [19].	Yes, activity reports are available with graphs for each module [15, 12].	Yes, See statistics of learner visits and the activities [6, 14].
Control Connections (tracking of learners)	Yes, module 'Tests and Surveys'	Yes, monitoring tools, statistics; + dashboard [19]	Yes, supervision tools (groups, activities, etc.); + dashboard.	Yes [14]
Report on the test results	Yes, 'tests and surveys' tool	No	Yes, visualization interface of the various reports of test results.	Yes [5]
Glossary	Yes, 'Glossary' tool	No	Yes, Glossary Module	Yes, Glossary Module [5]
Report on the frequency or the use of a course	Yes, 'My Tracker' tool	Yes, through a tracking interface	Yes, « Rapports d'activités » tool	Yes, 'Statistics' tool [5]
Functional appropriateness				
Certification	Yes [17]	Yes	Yes, plugin 'Certificate'	Yes
Foyer (family group)	Yes, creating a discussion forum	Yes, creating a discussion forum	Yes, discussion forum by user [15]	Yes, creating a discussion forum [5]
Registration chat	Yes [17, 18]	Yes	Yes, All sessions are recorded [15]	Yes [7]
Messaging	Yes, mail [17]	Yes, internal messaging [19]	Yes, messaging has existed since Moodle 1.6 [16].	Yes [7]
Plagiarism detection tools	No	No	Yes, plagiarism plugin 'Turnitin'	No
RSS / podcast: means of distributing files	No	Yes	Yes	Yes, Display the news contents via RSS [8]

Figure 4: Functional Suitability [Ouadoud et al. \[2016\]](#)

As we can see in Figure 4, Moodle has all but one functionality (management of the working time of learners and teachers), making it the most complete platform out of the four studied.

Finally, in regards to accessibility, [Moodle \[2023\]](#) specifies that Moodle first received WCAG 2.1 Level AA accreditation on 10 November 2020. Since this first accreditation, Moodle has undergone an annual audit and re-accreditation process.

2.3.2 Blackboard

Blackboard Learn is another popular learning management system (LMS) that is used by institutions around the world to create and manage online learning environments - Universidade do Minho is among the institutions that use Blackboard as its educational platform. Unlike Moodle, Blackboard Learn is proprietary and not open-source.

Due to its proprietary nature, it can be hard to find reliable literature that goes in-depth into its features, thus all the information displayed in the following paragraphs comes from their official help website, [Blackboard \[2023\]](#), and my personal experience as a student.

As an e-learning platform, it has a reputation for being reliable and stable and, on top of this, it offers a variety of tools and services, such as course design, analytics, reporting and many more, making it very appealing for large institutions that may rely on it or their online learning needs.

When it comes to the technologies used, there is a variety of them; however, looking at the code samples given as an example to developers, the core functionalities may use Java, HTML and JavaScript as their main programming languages, with MySQL serving as the default database.

Functionality-wise, it seems to have all the features a platform of this kind would need; among them are:

- Many tools and features enable educators to create and manage online courses, including activities, quizzes, assignments, resources, and forums.
- Support of various types of media, such as audio, video, and images, as well as file and document management.
- A built-in grade book that allows educators to track and record student progress and provides options for student self-assessment and feedback.
- A range of accessibility features, including support for screen readers and alternative languages.

Unfortunately, because I only had access to Blackboard as a student, it is hard to compare it with Moodle in more detail.

Finally, regarding accessibility, Blackboard Learn specifies that it follows the internationally recognized Web Content Accessibility Guidelines (WCAG) 2.1 Level AA as well as the section 508 standards in the United States.

Chapter 3

Architecture and Requirements

Use cases describe the system's functionalities from an end-user perspective, while the architecture outlines the technical design and structure of the system. This chapter describes the architecture followed by a more in-depth technical description of the project as well as a list of software requirements.

3.1 Project technical description

I have covered the motivation and objectives of this project but did not go into detail on its technical aspects.

As said before, Formare is a proprietary e-learning platform developed by Altice Labs. On the technical side, it is built on the .NET Framework 4.7.2 using Web Forms, which the company feels is a dated technology when it comes to developing and maintaining a project of this size.

The company was interested in modernizing the user interface because:

- Other teams inside the organization are using more contemporary options.
- The organization considers Web Forms an obsolete technology due to its lack of new features.
- Web Forms limits the possibility for a mobile application due to its architecture.
- The need for reactivity.
- The need for multi-language support.
- The possibility of using existing internal tools, developed for other projects, in Formare to ease development and maintenance.

Having said that, choosing the technology to be used on the front end was rather easy. The need for modern technology that allows for reactivity and multi-language support meant that any JS Framework

would fit the project. In the end, React¹ was the chosen library since other teams were already using it and had developed a styles library (using styled-components²) that would guarantee any accessibility and usability needs.

Unfortunately, the deeply coupled user interface and back office logic make it hard to easily add a new React client to the application. The most notorious problem with this scenario is connecting this new client to the existing business logic.

I did find a very simple working example of someone who managed to directly connect a react front-end to an MVC project³, however, the example was very simple and MVC works differently from Web Forms. Also, the technology was only experimental so I did not have the confidence that it would be good enough to support such a big project as Formare.

In the end, the best solution I could find was to introduce a web API into the project. Formare's back office used Web Forms features in some methods but those could be removed, making it easy to integrate with other .NET projects such as an ASP.NET Web API⁴ project. As for the .NET version I decided to go with the most recent LTS version (.NET 6).

Finally, I want to reiterate that Formare is a big project that has different websites depending on what kind of device and client is accessing the platform (desktop, mobile, etc.). It would be impossible for me to fully rework the sites; instead, I will work only on the website used by mobile devices.

3.2 Architecture

Before this project's introduction, Formare was built with ASP.NET Web Forms⁵.

¹ <https://react.dev/>

² <https://styled-components.com/>

³ <https://xabikos.com/2015/03/18/Using-Reactjs-net-in-Web-Forms>

⁴ <https://dotnet.microsoft.com/en-us/apps/aspnet/apis>

⁵ <https://learn.microsoft.com/en-us/aspnet/web-forms/>

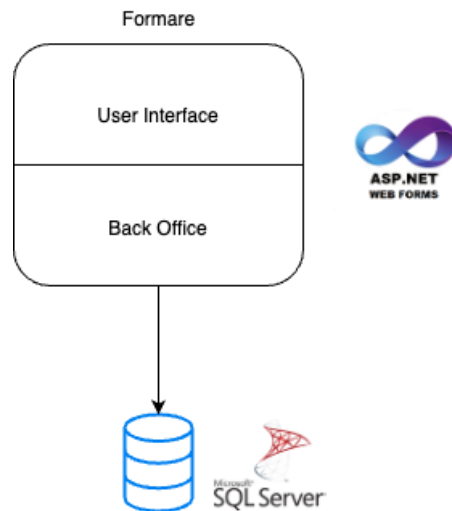


Figure 5: Previous architecture

As we can see in Figure 5, Formare's architecture was deeply coupled together. The user interface and the back office are both on the same project and it can be hard to distinguish the two, with some back office methods directly invoking user interface functionalities. This made it impossible to add a new front end to the existing application without making some changes to the existing architecture.

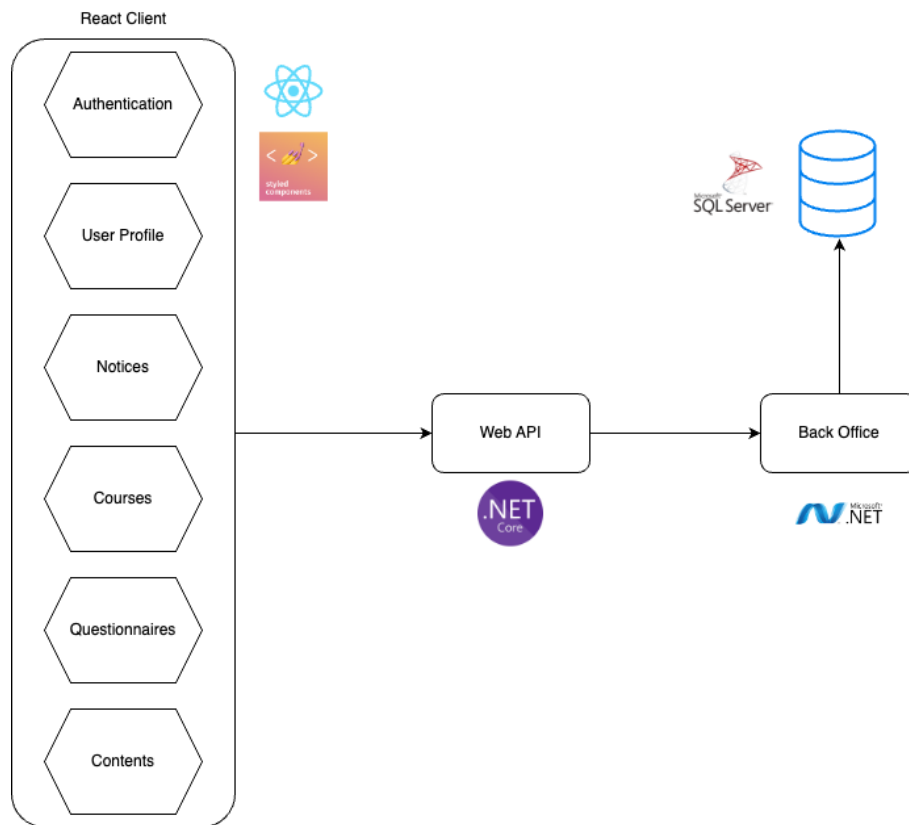


Figure 6: Current architecture

Figure 6 describes the current architecture. To the left side, we have the previous model where some changes have been made to the back office to separate it from the user interface and allow it to be used for purposes other than just serving a Web Forms application - some back office methods were calling Web Forms functionalities, so I updated and removed the user interface related elements.

To the right side of Figure 6, we can see the new front-end client and its different modules - authentication, user profile, notices, courses, questionnaires and contents - each with its own set of pages.

To allow communication between the new client and the updated back office, a web API was introduced. The front end makes HTTP requests to the API which in turn calls the needed back-office functionalities and sends an HTTP response back to the client.

As for the technologies used and seen in Figure 6, their choices will be clarified in the next section.

3.3 Requirements

Before starting implementation, a set of requirements was given to me by Altice Labs. Software requirements are paramount in developing robust and efficient software, therefore abiding by them was a major

priority during development.

3.3.1 Architecture software requirements

Number	Requirement
REQ-1	The new client must be built using the React library
REQ-2	The new client must preserve all existing functionalities from the original mobile website
REQ-3	A web API that mediates communication between client and back office must exist
REQ-4	The web API must convert the back office results into JSON and deliver them to the client via HTTP
REQ-5	Users must have a valid account and be able to log in using a username and password
REQ-6	All pages, except for the login, register and password recovery pages, should need authorization

Table 1: Architecture software requirements

3.3.2 Authentication software requirements

Number	Requirement
REQ-7	Upon successful authentication, a token with limited validity, compliant with system parameters, must be generated and sent to the client in all subsequent requests
REQ-8	Before authenticating a user, the system must validate the license
REQ-9	The authentication process must keep track of how many times a user has been authenticated
REQ-10	Upon successful authentication, the system should register the user access
REQ-11	Upon successful authentication, the user should be redirected to a new page displaying the list of courses they are enrolled in
REQ-12	In case of an invalid username, the error message should not reveal specific information about the error, only indicating that the credentials are incorrect
REQ-13	If the authentication fails due to an expired license, the error message should be based on the user's profile, providing specific information for administrators and more general guidance for other users, prompting them to contact an administrator

REQ-14	Upon logout, the authentication token should be deleted from the client and invalidated on the server
REQ-15	Upon logout, the user should be redirected to the login page

Table 2: Authentication software requirements

3.3.3 Account creation software requirements

Number	Requirement
REQ-16	The platform must provide users with the ability to create a new account, through the register page
REQ-17	The registration form should include the following inputs: name, username, email, password, and confirm password
REQ-18	A verification captcha must be implemented on the register page to prevent automated bot account creation
REQ-19	If the maximum number of available accounts allowed by the present license has been reached, the registration process must be denied
REQ-20	If the maximum number of available accounts allowed by the present license has been reached, the user should be informed that they need to contact an administrator
REQ-21	Upon successful registration, the user should be automatically redirected to the login page
REQ-22	Upon successful registration, the new account should be immediately accessible with the login credentials

Table 3: Account creation software requirements

3.3.4 Password recovery software requirements

Number	Requirement
REQ-23	Any registered user with a valid email address should be able to recover their password
REQ-24	The login page must contain a link directing the user to the password recovery page
REQ-25	The password recovery page should include an email input

REQ-26	Upon submitting a password recovery request, the server must validate the provided email address. If the email is valid, a new password should be generated and updated in the user's profile
REQ-27	Upon successful password recovery, the new password must be sent to the user's email
REQ-28	If the email address provided does not exist, a generic success message should be displayed, without revealing that the operation was unsuccessful
REQ-29	Only active accounts should be permitted to request a new password
REQ-30	If a request is made for an account that is suspended or archived, the user should be notified to contact an administrator

Table 4: Password recovery software requirements

3.3.5 Support for multi-language software requirements

Number	Requirement
REQ-31	The client must be implemented to support display in multiple languages
REQ-32	All client-only text should be capable of being presented in any of the supported languages
REQ-33	The default language for the client should be European Portuguese
REQ-34	The password recovery page should include an email input
REQ-35	The application should be designed to easily accommodate the addition of new language bindings in the future

Table 5: Support for multi-language software requirements

3.3.6 User profile related requirements

Number	Requirement
REQ-36	Every user must have access to their respective profile page
REQ-37	On the profile page, users should be able to view their data for verification purposes

REQ-38	Users must be able to update their data by using a form similar to the registration form, excluding the captcha and any password-specific inputs
REQ-39	To update their password, users should navigate to a designated page with a form that includes a password input and a password confirmation input

Table 6: User profile software requirements

3.3.7 List of courses software requirements

Number	Requirement
REQ-40	The list of courses page must display essential information for each course, including the course name, start date, and finish date
REQ-41	If the course has a ranking property, the user's ranking should also be displayed
REQ-42	Users should be able to click on a course item from the list to access the course's main menu

Table 7: List of courses software requirements

3.3.8 List of notices software requirements

Number	Requirement
REQ-43	The platform should include a dedicated page for displaying recently posted notices
REQ-44	The notices page should display all notices, including system-wide and course-specific notices, relevant to logged-in user
REQ-45	Users should be able to click on a notice to view details such as the title, body text, images, and associated files
REQ-46	The page should display the number of unread notices at the top
REQ-47	When a notice is opened, the count of unread notices should be updated accordingly

Table 8: List of notices software requirements

3.3.9 List of contents software requirements

Number	Requirement
REQ-48	The course menu page should feature an option labelled "Contents" that leads to a new page listing all the training contents within the course
REQ-49	Contents may be categorized into different types - links, files and multimedia
REQ-50	Expired content should be visible on the contents page but inaccessible to users
REQ-51	Clicking on an item should redirect the user to a new page displaying key details about the selected content, along with a button to open it

Table 9: List of contents software requirements

3.3.10 List of questionnaires software requirements

Number	Requirement
REQ-52	The course menu page should include an option labelled "Questionnaires" that redirects the user to a new page listing all the questionnaires within the course
REQ-53	Each listed questionnaire should display information such as name, date limit, the number of attempts made, the maximum number of attempts allowed, and the current score
REQ-54	Expired questionnaires and those with the maximum number of attempts reached should remain visible on the list but be inaccessible to the users
REQ-55	A clear message explaining why the questionnaire is inaccessible should be displayed to the user
REQ-56	Clicking on a questionnaire should redirect the user to either an instructions page or the questionnaire itself, depending on the questionnaire settings

Table 10: List of questionnaires software requirements

3.3.11 Questionnaire instructions software requirements

Number	Requirement
REQ-57	When a questionnaire that is set up to show instructions is clicked, the user should be redirected to the instructions page

REQ-58	The instructions page must include the questionnaire time limit and consequences of the timeout
REQ-58	The instructions page must include the questionnaire minimal score required
REQ-59	The instructions page must specify if the user is required to answer all questions before submitting the questionnaire
REQ-60	The instructions page must let the user know what he will see after submitting his answers
REQ-61	The instructions page must include the questionnaire time limit and consequences of the timeout
REQ-62	The instructions page must specify if the user is allowed to save his answers and return later to the questionnaire
REQ-63	The instructions page must clarify which submission will be used for scoring
REQ-64	The instructions page must include any additional information the questionnaire creator wishes to include
REQ-65	At the end of the instructions page, a button should be provided to redirect the user to the actual questionnaire

Table 11: Questionnaire instructions software requirements

3.3.12 Answering and submitting questionnaires software requirements

Number	Requirement
REQ-66	The questionnaire may include multiple types of questions - multiple-choice, multiple-answer and true or false
REQ-67	The user's selected answers and subsequent submission must be saved in the application
REQ-68	The user's grade must also be saved and associated with their submission
REQ-69	The questionnaire page should have the following buttons - "Submit" button, "Abandon" button and "Save Answers" button
REQ-70	Upon submitting the questionnaire, the user should be redirected to a new page where they can view their submission and any corresponding feedback or grades

REQ-71	If a questionnaire has a time limit, a timer should be displayed during the questionnaire
--------	---

Table 12: Answering and submitting questionnaires software requirements

Chapter 4

Use Cases and Implementation

4.1 Integration

Having now decided to have a server in between the client and the application's logic layer, I must find ways to allow the .NET 6 server to use the services built on the back office even though they are written in different versions of the framework. Here I will describe some of the problems I found that do not belong in any particular use case.

4.1.1 Configuration files

Configuration was the first problem I found when integrating the back office with the API.

While the .NET Framework uses XML files for settings and configuration, .NET 6 uses JSON. This is problematic because whenever I call a back office method, the code looks for XML but the API uses JSON. I could keep two configuration files in different formats but if the files somehow differ, unexpected behaviour will happen.

The solution was to abandon one of the formats. Abandoning the JSON files was the easier option since only the API was using them. This was done by installing the package `System.Configuration.ConfigurationManager`¹ which supports the use of XML configuration files.

4.1.2 Injecting the services

The second problem I found was with service injection. In .NET 6, services require constructors to initialise properties, otherwise, they will throw exceptions when being built. However, no constructors are used in the back office due to the usage of the Unity Framework²; one can instead use the `[Dependency]` attribute

¹ <https://www.nuget.org/packages/System.Configuration.ConfigurationManager>

² <https://www.developer.com/design/microsoft-unity-framework-a-peek/>

on the injected properties.

The fix here is to add a constructor that initialises all needed properties in every back-office service that needs to be injected in the API.

4.1.3 Default implementations

Another recurring problem is how some of the default implementations differ from .NET 6 to .NET Framework.

The most notorious example is the different `Encoding.Default` values between the two versions. In .NET Framework (which only runs on Windows machines), this property returns the system's encoding. In .NET 6, on the other hand, it always returns `UTF-8`, regardless of the operating system it's running on³. This has caused problems every time the back office needs to process some kind of encoded data, such as passwords. The solution is to always specify what encoding is being used, so, instead of writing `Encoding.Default`, we write `Encoding.UTF-8`.

4.1.4 Requests and responses

Another problem that commonly came up was how to structure JSON requests and responses. Previously, the application could simply access the user input and perform actions according to it without having to structure an API call. Now, however, there is a strict separation between the front end and the back end and the only way in which they can communicate is through HTTP calls. This means that JSON requests and responses must be carefully designed to re-use the business logic without too many alterations.

In the Web Forms application, when an error occurred the view code could be altered right after the error to show a corresponding message. That was possible because the view and business logic code were running together on the page. This workflow is not possible in the new mobile web application.

An error is a JSON response that needs to be parsed; as such, error responses should all follow the same structure so they can be easily read. For this reason, I am using the problem details⁴ specification.

When it comes to successful responses, it is hard to specify a pattern for them as they, unlike errors, depend on the route and what the route does. After all, it is unlikely a login request and a request to get all courses return the same data.

To easily distinguish successful responses and error responses, as well as access their data with the correct type definitions, I created a custom hook that fetches data and parses it. I also added the

³ <https://learn.microsoft.com/en-us/dotnet/api/system.text.encoding.default?view=net-6.0>

⁴ <https://datatracker.ietf.org/doc/html/rfc7807>

functionality to perform specific actions depending on the request status. I based the functionality and interface on the popular javascript library `tanstack-query`⁵

```
const response = useFetch(makeRequestMethod, {
  // If the request is successful this happens,
  onData: () => ...
  // If the request is not successful this happens,
  onError: () => ...
  // If true runs immediately
  // If false runs only when the "runFetch" method is called
  runFetchNow: false
});

response.runFetch(...);
```

4.1.5 Distinguishing between client and server logic

Another common pitfall is understanding what should run on the client and what should run on the server. A common example is form validation. If the user has to enter an email address somewhere in a form where should the application validate it, in the server or the client?

On a client-server project, the server should always do some sort of validation of its own due to security reasons, but what about the client? Should the client do any kind of validation since the server is already doing those?

In the end, the client should validate everything and the server only the critical parts. Forms are validated on the React application using an in-house library and, only if those validations are successful, is the request sent. Then the back end can perform its own set of validations.

Some needed validation could not be done using the in-house library, so I added the functionality to validate input to my custom hook.

```
response.runFetch({
  // The request arguments (route parameters and request body)
  args: [params, { ... }],
  // A list of tests - [{ test: boolean }]
```

⁵ <https://tanstack.com/query/latest/docs/react/overview>

```

// If any is false:
// The request is cancelled and an error is returned
validations: [{ doPasswordsMatch: confirmPass == actualPass }]
});

```

4.2 Use cases

Having now gone into a detailed description of the project, technologies used, and general problems found, I will describe the use cases followed during implementation and what problems and solutions I found for each.

4.2.1 Authenticate

Use Case Name	Authenticate
Description	Authenticate in the application to access the restricted pages
Actors/Roles	Any user
Pre-condition	None
Trigger	The user fills out a login form with their credentials and then submits it
Post-condition	The user is authenticated. The application front-end receives a cookie verifying the user's identity
Level	User
Main Success Scenarios	1. The application validates the entered credentials against the saved data in the database 2. The application validates the licence 3. An authentication cookie is generated 4. The login is registered in the database 5. The user is redirected to their home page

Extensions	1. The username and the password don't match or the username doesn't exist 1.1. An error message is shown indicating that the credentials are invalid. At no point should the user be made aware of the existence (or lack thereof) of the username in the system 2. The licence has expired 2.1. If the user is an administrator, an error message is shown indicating that the licence has expired and that they should access the desktop version to update it 2.2. If the user is not an administrator, an error message is shown indicating that an error has occurred and that they should contact an administrator
------------	---

Table 13: Authenticate use case

This was the first use case I tackled during this project's development, therefore, there were some added difficulties. The first problem was the fact that I was new to the code base and was still getting accustomed to the new technologies I was using. The second hardship was understanding how to re-write functionality in a new architecture, specifically how to structure the requests and the responses.

When it came to authentication-related problems, two immediately stood out.

The first, simpler one, is about licence registration and how the server can tell the client if the licence is valid, if it is about to expire, or if already expired. This was confusing initially because the authenticate response could return OK even if the licence was about to expire, in which case the user should be warned. The implemented solution was to add a custom header to the response with the licence status.

The second, more complex problem, was how the server could authenticate the user. The old platform was using Forms Authentication⁶ and its own authentication system. The problem here is how can I bring this system over to a web API. After researching the topic I realized that the back-office authentication system and the Forms Authentication were independent of each other, the latter serving mostly as a container for the information provided by the former. That meant that I could use authentication cookies on the web API and create a list of claims that contained whatever important information about the authenticated user the back office provided.

```
// Whatever info the cookie needs is saved on the claims
```

⁶ <https://support.microsoft.com/en-au/topic/understanding-the-forms-authentication-ticket-and-cookie-8ff63703-d7e0-5282-b8a6-9d516ee8a04d>

```
// ex: username and id
var claims = new List<Claims> { ... };
var identity = new ClaimsIdentity(claims, "cookie name");
var principal = new ClaimsPrincipal(identity);

// Sign in the user
await HttpContext.SignInAsync(
    principal,
    new AuthenticationProperties { ... });
```

4.2.2 Account creation

Use Case Name	Account creation
Description	Create a new account in the application
Actors/Roles	Any user
Pre-condition	None
Trigger	The user fills out a registration form with his credentials and then submits it
Post-condition	The user is sent back to the main page
Level	User
Main Success Scenarios	1. The application creates a new, active, account
Extensions	1. The number of authorised accounts has been exceeded 1.1. The user shall receive a message telling them to contact the application's administrator

Table 14: Account creation use case

Account creation is similar to the authenticate use case, so one would think that there would be no particular problems with this use case; however, some aspects of the back office code proved to be hard to integrate.

As I previously mentioned, Formare has its own system for authentication. This system is not unique to the e-learning platform and is used by other platforms on Altice Labs, which means that I did not have access to its source code and could not modify it to fit my needs.

For some unknown reason, invoking the system's account-creating methods from the API was causing it to break and behave unexpectedly. In the end, after sharing my concerns with the Formare team, they decided to re-write the authentication code directly in the back office, allowing much more freedom in adapting the code to work on the API. This re-implementation ended up fixing the problem altogether.

On the front end, despite most of the code from the login page being reusable on the registration page, there was one key difference: the registration page should also have a verification step, to verify the user is not a bot.

In the old project, this verification code is generated manually and readily shown to the user, however, this would not be the case for the API since this code only worked on Windows while the API was intended to be cross-platform, and the team was interested in moving away from it in favour of a more modern approach such as Google reCAPTCHA V2⁷. This was easily accomplished by using the npm package `react-google-recaptcha`.

```
import ReCAPTCHA from "react-google-recaptcha";
<ReCAPTCHA
  sitekey={...} // The site key given by Google reCAPTCHA
  hl={...} // The language the captcha should appear in
/>
```

4.2.3 Logout

Use Case Name	Logout
Description	Terminate the current session in the application
Actors/Roles	Any user
Pre-condition	The user is logged in
Trigger	The user clicks on the logout button on the main menu
Post-condition	The user is unauthenticated and sent to the login page
Level	User
Main Success Scenarios	1. The application invalidates the current token 2. The user is redirected to the login page
Extensions	None

⁷ https://developers.google.com/recaptcha/docs/versions?hl=en#recaptcha_v2_im_not_a_robot_checkbox

Table 15: Logout use case

When the authenticate use case was implemented, the cookie generation was handled by ASP.NET Core cookie authentication. This makes it easy to implement the logout use case, the only step needed is to let ASP.NET Core handle it by invoking the `SignOutAsync` method.

```
await HttpContext.SignOutAsync("cookie name");
```

4.2.4 Password recovery

Use Case Name	Password recovery
Description	Obtain a new password to access the application
Actors/Roles	Any user
Pre-condition	None
Trigger	The user fills out the password recovery form and submits it
Post-condition	The user is sent a new password
Level	User
Main Success Scenarios	1. The application validates the entered e-mail address against the saved data in the database 2. The application validates the status of the account (only active accounts will receive a new password) 3. A new password is generated and saved 4. The new password is e-mailed to the user
Extensions	1. The e-mail address does not exist 1.1. The application shall not inform the user of that fact, acting as if the operation succeeded 2. The user's account is not active 2.1. The user is told to contact an administrator

Table 16: Password recovery use case

Regarding the front end, this was the simplest so far to implement, with a single email input form being needed.

The back end, on the other hand, had some added difficulties due to Formare's notification system. This system could not be injected properly into the API because it was using the Microsoft Unity Framework, which has been deprecated. Thankfully this system was in the middle of being renewed and when the reimplementations was finished everything worked as expected.

4.2.5 Profile

Use Case Name	Profile
Description	View and edit one's personal information and settings
Actors/Roles	Any user
Pre-condition	The user is logged in
Trigger	The user clicks on the "Profile" button on the main menu
Post-condition	The user stays on the same page
Level	User
Main Success Scenarios	1. The user's data are updated
Extensions	None

Table 17: Profile use case

In the old application, the user profile page is divided into two pages. One where the user can edit their name and email, as well as see their username and identity number and another where the user can change the password.

At first, the idea was to consolidate these two pages into one on the new mobile web platform, but doing so would mean re-writing a lot of back office code. This was due to the much more complicated form validation on the front end and more intricate business logic on the server.

In the end, it was decided that these two pages would remain separate so that the back office could be used without modification.

4.2.6 Multi-language

Use Case Name	Multi-language
Description	The application interface should be available in multiple languages
Actors/Roles	Any user

Pre-condition	None
Trigger	The user opens any page in the application
Post-condition	The front end renders in whichever language the user prefers
Level	User
Main Success Scenarios	1. The application uses the default language (European Portuguese) 2. The application uses the user-selected language
Extensions	None

Table 18: Multi-language use case

The in-house component library used during this project had some custom hooks that allowed me to easily implement this use case. All that was needed was to declare an interface with the possible message types (most, if not all, are strings) and an equivalent JSON file with the messages in their specific language.

```
// The language interface file:
export interface ILanguage {
  username: string;
  password: string;
}

// The JSON pt-pt file
{
  "username": "Nome de utilizador",
  "password": "Palavra-passe"
}
```

Now, whenever I want the application to show the username string in the correct language I need to use the in-house library hook `useSelector`.

```
const language = useSelector(...);
language.username // returns Nome de utilizador
```

I also created a hook to translate the error messages that come from the back-end responses. The errors are in the problem details format, therefore they all have a title. The hook looks at the title and compares it to a list of possible matches and decides which message should be shown to the user.

```
const { translateProblemDetails } = useLanguage();
```

```

errorTitle = "User not found";

// returns "Utilizador não encontrado"
translateProblemDetails(errorTitle)

```

4.2.7 List of courses

Use Case Name	List of courses
Description	View the list of courses the user is registered in
Actors/Roles	Any user
Pre-condition	The user is signed in
Trigger	The user enters the courses page
Post-condition	The list of courses the user is registered in is shown to the user
Level	User
Main Success Scenarios	1. The application shows the courses the user is registered in 2. The application tells the user they are not registered in any course at the moment
Extensions	None

Table 19: List of courses use case

When it comes to the front end, this use case was very straightforward to implement. All that was needed was a request to the server to fetch the courses and then render them to the user, using the components in the proprietary component library.

The back end, on the other hand, had some unexpected complications due to the data type returned by the back office when requesting the list of courses. The back office returned the list of courses in `DataTable`, which was very useful on the old application since these objects allow the developers to easily render their contents to the user in web forms applications. On the API, however, this feature becomes hollow and the server needs to parse the table before sending it to the client.

A package named `Json.NET`⁸ came in handy since it provided the feature the server needed, converting `DataTables` to JSON strings.

```
// We convert the dataTable into JSON
```

⁸ <https://www.nuget.org/packages/Newtonsoft.Json>

```
var json = JsonConvert.SerializeObject(dataTable);
```

4.2.8 Course menu

Use Case Name	Course menu
Description	The menu page where the user chooses what to do inside the course
Actors/Roles	Any user
Pre-condition	The user is signed in
Trigger	The user enters the course menu page
Post-condition	Opens a page with links to other pages inside the selected course (training contents, notices, quizzes, etc.)
Level	User
Main Success Scenarios	1. The application menu page is shown to the user
Extensions	None

Table 20: Course menu use case

This use case involves only the client and all it required was adding a new route nested inside the course route with a list of options the user could choose from. Each of these options would redirect them to a new page.

4.2.9 List of notices

Use Case Name	List of notices
Description	View the list of notices (general and per course) destined for the user
Actors/Roles	Any user
Pre-condition	The user is signed in
Trigger	The user enters the notices page
Post-condition	A list of all notices is shown to the user
Level	User
Main Success Scenarios	1. The application shows the notices to the user 2. The application tells the user there are no notices

Extensions	None
------------	------

Table 21: List of notices use case

While the “list of courses” use case merely required me to fetch minimal information about each course and show it to the user so he can select where to go next, notices must be fetched in full and their purpose is not to help the user navigate the platform but to show information. A notice can have files, images and any kind of text; they can also be read or unread.

This use case needs two different routes, one for fetching notices and another for marking them as read by the signed-in user.

Starting with the latter, marking notices as read was very straightforward, needing almost no modifications to the old code.

The former was more complicated. On the old application, a list of notices would be fetched, with each entry on the list having just the key data. When a notice was clicked, the remaining information would be given by the back office and shown to the user. This two-step process could not be re-enacted on the new platform since the component library used did not have the necessary components. This left me with two options: redirecting the user upon clicking a notice to a new page where all information about the selected notice would be shown; or creating a route that would return all the information of each notice, instead of dividing this process into two steps (first fetching only the key info and then fetching everything for each notice) as the old platform did.

This behaviour could not be implemented in the new platform since the component library used did not have the necessary components; also, the team wanted the list of notices and the info to be all on the same page, which rendered the option to redirect the user upon clicking a notice to a new page with all the information about the selected notice useless. The team preferred the notices to be all on one page, making the first option an unacceptable solution.

```
// A list with only the key info about each notice
// Key info is Id, Title, Date, ...
var minimalNoticesList = ...

// For each notice we get the remaining information (content)
// and create a new list
var fullNoticesList = minimalNoticesList.Select(min => {
    var currentNotice = new Notice(min.Id)
```

```
return new Object(min, currentNotice);
});
```

Going to the client now, on page load, the notices are fetched and rendered; when a notice is clicked, the app verifies it is already marked as read; if not, the server is notified and updates it. Also, the front-end component used for each notice is similar to an accordion component, so when clicked, the full content of the notice is shown. This content comes in HTML format, so to render it I used the `html-react-parser`⁹ package. Notices could also come with images and files.

```
// Files
<a
  href={`data${type of content};base64;{file}`}
  download />

// Images
<img
  src={`data${type of content};base64;{image}`}
  alt={title} />
```

4.2.10 List of contents

Use Case Name	List of contents
Description	View the list of training contents that a selected course has
Actors/Roles	Any user
Pre-condition	The user is logged in and enrolled in the class
Trigger	The user enters the contents page
Post-condition	A list of all contents belonging to the selected course is shown to the user
Level	User
Main Success Scenarios	1. The application shows the contents to the user 2. The application tells the user there are no contents to access
Extensions	None

Table 22: List of contents use case

⁹ <https://github.com/remarkablemark/html-react-parser>

This use case was very straightforward since it is very similar to the list of notices use case. On the front end, most of the code could be reused since the behaviour for both pages is mostly the same.

The same applies to the backend, transforming DataTables to the corresponding JSON string.

4.2.11 Display content of type link

Use Case Name	Display content of type link
Description	Contents of type “link” are displayed to the user
Actors/Roles	Any user
Pre-condition	The user is logged in and enrolled in the class
Trigger	The user enters the details page for a content of type “link”
Post-condition	The browser opens a new tab containing the page the link points to
Level	User
Main Success Scenarios	1. The browser opens a new tab containing the page the link points to
Extensions	None

Table 23: Display content of type link use case

It is important to understand, for both this and the following use cases, that there are different types of learning contents (link, file, non-standardised and standardised).

When a user clicks on content, on the list of contents page, he is redirected to a page with more details about the content he clicked. On that page, there is also a button that opens the content. This button behaves differently depending on the type of content.

Links are perhaps the easier ones to implement. On the server side, all that is needed is to search the database for the selected content, given the course and content IDs, and send it to the client.

On the client, we show the user the key details of the requested content and render a button at the bottom of the page that allows the user to open a new tab with the page that the link refers to.

4.2.12 Display content of type file

Use Case Name	Display content of type file
Description	Contents of type “file” are displayed to the user

Actors/Roles	Any user
Pre-condition	The user is logged in and enrolled in the class
Trigger	The user enters the details page for a content of type “file”
Post-condition	The user can download the file embedded in the content
Level	User
Main Success Scenarios	1. The user downloads the file
Extensions	None

Table 24: Display content of type file use case

Files require further work on both the back end and front end.

On the server, there is now a new route that queries the metadata belonging to the file and sends it to the client via HTTP using the ASP.NET return function `File`.

Now, on the client, when the button is clicked, it requests the new route and the file is downloaded.

4.2.13 Display content of type non-standardised

Use Case Name	Display content of type non-standardised
Description	Contents of type “non-standardised” are displayed to the user
Actors/Roles	Any user
Pre-condition	The user is logged in and enrolled in the class
Trigger	The user enters the details page for the content of type “non-standardised”
Post-condition	The user can open the content directly on the page
Level	User
Main Success Scenarios	1. The user sees the content
Extensions	None

Table 25: Display content of type non-standardised use case

Non-standardised training contents are composed of HTML pages that are to be rendered on the page rather than downloaded – a slide show, for example. This is implemented using an `iframe` tag with a source pointing to the content repository.

There is now a new route on the server that searches for the path to the starting file of the content (index.html).

On the client, once again we can re-use most of the code, only changing the button's behaviour. When clicked, the client fetches the path to the content and renders it using an iframe tag.

As one can see, this use case is fairly simple. There is, however, one detail that when overlooked, breaks the whole process. Where is the content hosted?

The server returns a path that points to where the content is, but the client won't accept content from anywhere; after all, modern browsers do not allow local resources to be rendered on the page due to security concerns unless those resources are in the public folder, which unfortunately, they are not. To make sure everything works perfectly together, a virtual directory was created to conceal the physical directory where the contents are.

4.2.14 List of questionnaires

Use Case Name	List of questionnaires
Description	Display a list of questionnaires from a specific course
Actors/Roles	Any user
Pre-condition	The user is logged in and enrolled in the class
Trigger	The user enters the questionnaires page
Post-condition	The list of questionnaires is displayed on the page
Level	User
Main Success Scenarios	1. The user sees the list
Extensions	None

Table 26: List of questionnaires use case

This was the first use case I tackled when implementing the questionnaire service.

This use case is similar to other list-related functionalities, therefore most of the code can be reused.

4.2.15 Display questionnaire

Use Case Name	Display questionnaire
Description	Display the questions for the selected questionnaire

Actors/Roles	Any user
Pre-condition	The user is logged in and enrolled in the class
Trigger	The user clicks the questionnaire they want to access
Post-condition	The user can interact with the questionnaire
Level	User
Main Success Scenarios	1. The user is now on the questionnaire page
Extensions	None

Table 27: Display questionnaire use case

After successfully viewing the available questionnaires in the classroom, the user now needs to be able to select one and see its questions. This use case is a very complex one, that must accommodate different behaviours depending on the questionnaire configuration. Let's look at those behaviours one at a time.

Firstly, when clicking on a questionnaire, one of two things can happen depending on how the questionnaire was set up by its creator: the user is redirected to the questionnaire page; or the user is redirected to an instructions page, where information about how the test works is shown as well as a button to redirect the user to the actual questionnaire.

To implement this behaviour I created two new routes on the server: one that returns the questionnaire instructions as well as a field that indicates to the client if these instructions should be rendered or ignored and a route that returns the questionnaire itself (questions, previous answers, percentages, etc...).

On the client, upon clicking on a questionnaire, the user is always redirected to the instructions page, where the application decides if the user should be redirected once again to the page with the test or if the instructions received from the server should be shown to the user.

```
const instructions = useFetch(makeInstructionsRequestMethod, {
  fetchArgs: // request arguments,
  onData: (data) => {
    if (data.showInstructions)
      // stay on the page and render the instructions
    else
      // redirect the user to the test page
  }
})
```

```
});
```

Now it is time to render the questions. As I said previously on the requirements, there are three different types of questions: multiple choice, where there are several options and the user can only select one; multiple answers, where there are several options and the user can select as many as they want; and true or false, where there are only two options (true and false) and the user can select only one.

Since each question object comes with a field that indicates its type, the client just needs to look at this field and render each question accordingly.

```
{questions.map(item => {  
  if (item.type == "multiple choice")  
    return // multiple choice component  
  else if (item.type == "multiple answers")  
    return // multiple answers component  
  else if (item.type == "true or false")  
    return // true or false component  
})}
```

Finally, some visuals can appear or not on the page. One example is question percentages – depending on the settings, these can be shown or not to the user during the questionnaire resolution. React does allow one to render things conditionally, so the solution to this problem came down to properly using that feature.

4.2.16 Save questionnaire answers

Use Case Name	Save questionnaire answers
Description	The user saves the current state of his answers to the ongoing questionnaire
Actors/Roles	Any user
Pre-condition	The user is logged in and enrolled in the class, and the questionnaire options are set to allow the user to save and finish later
Trigger	The user clicks the “save questionnaire” button
Post-condition	The user’s answers are saved to the database and he can come back later to finish his submission

Level	User
Main Success Scenarios	1. The user is redirected to the questionnaire list page
Extensions	None

Table 28: Save questionnaire answers use case

Another behaviour that can change from questionnaire to questionnaire is the user's ability to save his answers and come back later to finish his test.

On questionnaires that support this feature, there should be a button that, when clicked, should save the user's answers and redirect him to the questionnaire list page.

The implementation itself for this use case was very straightforward, the only problem being that not all questionnaires allow the user to save their answers. The solution, once again, came down to using React's ability to render components conditionally.

```
{questionnaire.saveAndFinishLater &&
  // Render "Save button"
}
```

4.2.17 Submit questionnaire answers

Use Case Name	Submit questionnaire answers
Description	The user submits their answers to the current questionnaire
Actors/Roles	Any user
Pre-condition	The user is logged in, enrolled in the class, and is currently answering a questionnaire
Trigger	The user clicks the "submit questionnaire" button
Post-condition	The user's answers are submitted and the questionnaire is finished
Level	User
Main Success Scenarios	1. The user is redirected to either the list of questionnaires or the questionnaire answers page, depending on the options set in the questionnaire
Extensions	None

Table 29: Submit questionnaire answers use case

The submit questionnaire use case was perhaps the most complex use case of the questionnaire-related ones.

The front end was simple, having most of the code already written for the save questionnaire functionality since they behave the same way on the client.

It was the server that proved to be a challenge. The server logic for this use case can be divided into two different aspects.

The first one involves saving the answers to the database, which is identical to the code used for saving the answers.

The second is when the questionnaire is submitted. The score is calculated and several different things can happen depending on how the questionnaire was set up. The problem here is that the method built to run this logic was built under the preconception that the code would be run on a web forms application and therefore the server would throw an exception whenever this method was called from the API. The solution was to rewrite this method and its logic in ways that are more appropriate for a web API - instead of forcefully redirecting the user to a new page, a response is sent with any necessary information so the client can react accordingly.

```
// Method used in a web forms application
if (...)
{
    // Method that redirects the user to a new page
    Redirect("some page")
}

// Method used in the API
if (...)
{
    // Two variables are created and returned later to the
    // client where the user will be redirected
    var status = 0;
    var message = "SUCCESS";
}
```

4.2.18 Abandon questionnaire

Use Case Name	Abandon questionnaire
Description	The user abandons the ongoing questionnaire
Actors/Roles	Any user
Pre-condition	The user is logged in, enrolled in the class, and currently solving the questionnaire
Trigger	The user clicks on the “abandon questionnaire” button
Post-condition	The user’s current questionnaire session is deleted
Level	User
Main Success Scenarios	1. The user is redirected to the questionnaire list page
Extensions	None

Table 30: Abandon questionnaire use case

The abandon questionnaire use case was very straightforward both on the server and on the client.

The current session is abandoned and, if the questionnaire settings allow the user to save their answers, any previously saved session is removed.

On the client, a button was created to allow the user to leave the questionnaire page, upon clicking this button a modal is opened so the user can confirm his action.

Chapter 5

Final Results

The preceding chapters delved into the architecture, requirements, and use cases that underpin the development process. Now, I discuss how the platform aligns with the initially set objectives, shedding light on its strengths and limitations.

5.1 Authentication and profile

Looking at the authentication and profile features, the final result successfully emulated the previous implementation behaviour. If a user came from the previous interface to the new one he would know how to log in, register and change their account information.

As for improvements, users who are authenticated and go to the login page URL are now redirected to the login page, styles no longer fail to render, which was a common occurrence on the Web Forms platform and both pages are now responsive making them usable in devices of any size.

Figure 7 shows the new login and profile pages.



(a) Login page

(b) Profile page

Figure 7: Authentication and profile pages

All software requirements regarding authentication, account creation, password recovery and user profile were fulfilled.

5.2 Courses

Both the list of courses and the course menu pages were very straightforward in functionality - their only use was to display information about which courses the user can access and what can he do in each course.

Once again, the original behaviour was maintained to accommodate long-time users. The number of reloads and redirections was also improved, since now upon clicking on one course the page is swiped and the list of actions inside the course is shown to the user, as opposed to reloading and redirecting to a new page.

Figure 8 shows the new courses page.

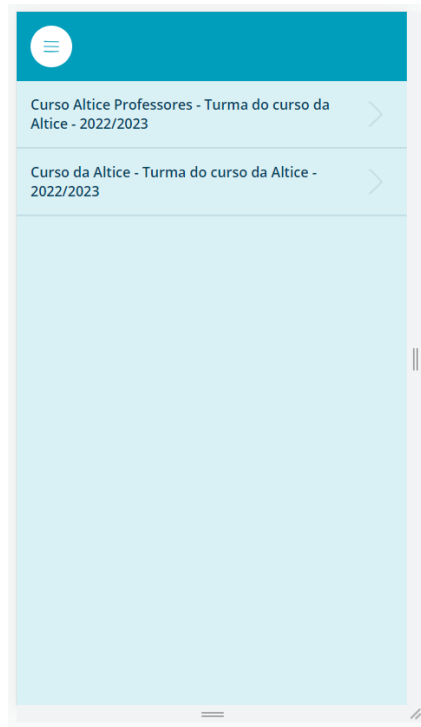


Figure 8: Courses page

All software requirements regarding courses were fulfilled.

5.3 Notices

Notices were also a very straightforward feature. In the previous platform, the user entered a page and saw a list of notices. Upon clicking on one the notice would expand downwards and he would see its details.

As per usual, original behaviour was maintained to accommodate long-time users. The visuals were significantly improved as well as accessibility and usability since the older page had small expand buttons. On the new page, using the react styles library, the buttons now follow accessibility and usability good practices making them easier to click in small size devices.

Another point of improvement is when updating the unread notices number at the top of the page after clicking a notice (it should decrease). React does this effortlessly by manipulating the state while the older application uses a very complicated method. In the project maintainability was improved.

All software requirements regarding notices were fulfilled.

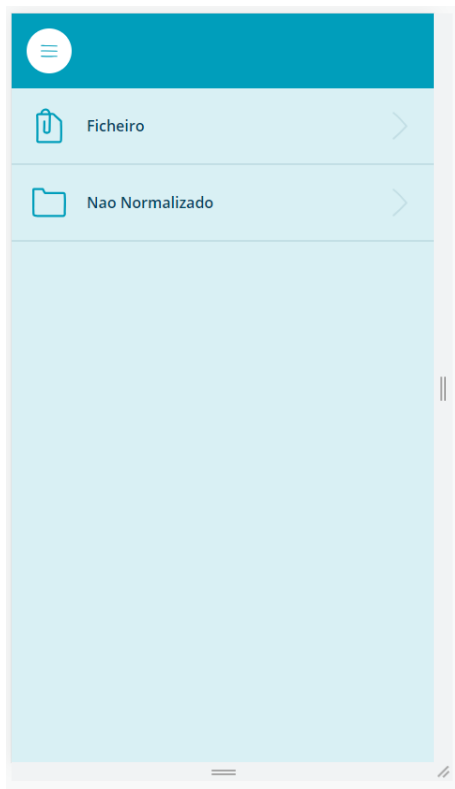
5.4 Content

Content-related functionalities had varying degrees of complexity, and that complexity, together with some time constraints, made it impossible to implement every behaviour present on the original platform. The change from a monolithic architecture to a client-server one as well as the need to use the existing in-house component library also meant that some behaviours had to be modified to better fit the new approach.

Whilst listing the content kept most of the old behaviour, again to accommodate long-time users, only updating the visuals, some changes had to be introduced when displaying the content to the user due to the migration from Web Forms to a web API. Despite these changes, users coming from the old platform will still know how to view different types of content.

Improvements were made to the navigation of contents. On the old platform clicking on content should redirect the user to a new page or in some cases open the content directly on the page, however, this behaviour was buggy and uncertain. The new implementation fixes all these problems and gives even more information to the user about the content he is accessing by always redirecting him to a new page with the content and some important information.

Figure 9 shows the new list of contents and content pages.



(a) List of contents page



(b) Content page

Figure 9: Content related pages

All software requirements regarding contents were fulfilled apart from the ones shown in Table 31:

Number	Requirement
REQ-51	Clicking on an item should redirect the user to a new page displaying key details about the selected content, along with a button to open it

Table 31: Unfulfilled contents software requirements

This requirement was not fulfilled because standardised contents were not implemented during development due to time and complexity constraints.

5.5 Questionnaires

Just as it was with the content-related functionalities, not all questionnaire-related behaviours could be implemented due to time constraints.

Once again, the end user will not sense any differences in usability – apart from the visual uplift –

since most changes were made on the server side. It is important to note that all changes made to the server were made not to improve the behaviour but to maintain it, adapting it to the web API.

Questionnaires are very complex and involve a vast list of services to work together and therefore changing the code unnecessarily could introduce new bugs. All improvements were made on the client, the most notable one being introducing a new modal component upon clicking the “abandon questionnaire” button to confirm the user’s intention to leave the test.

Figure 10 shows the new questionnaires and questionnaire pages.

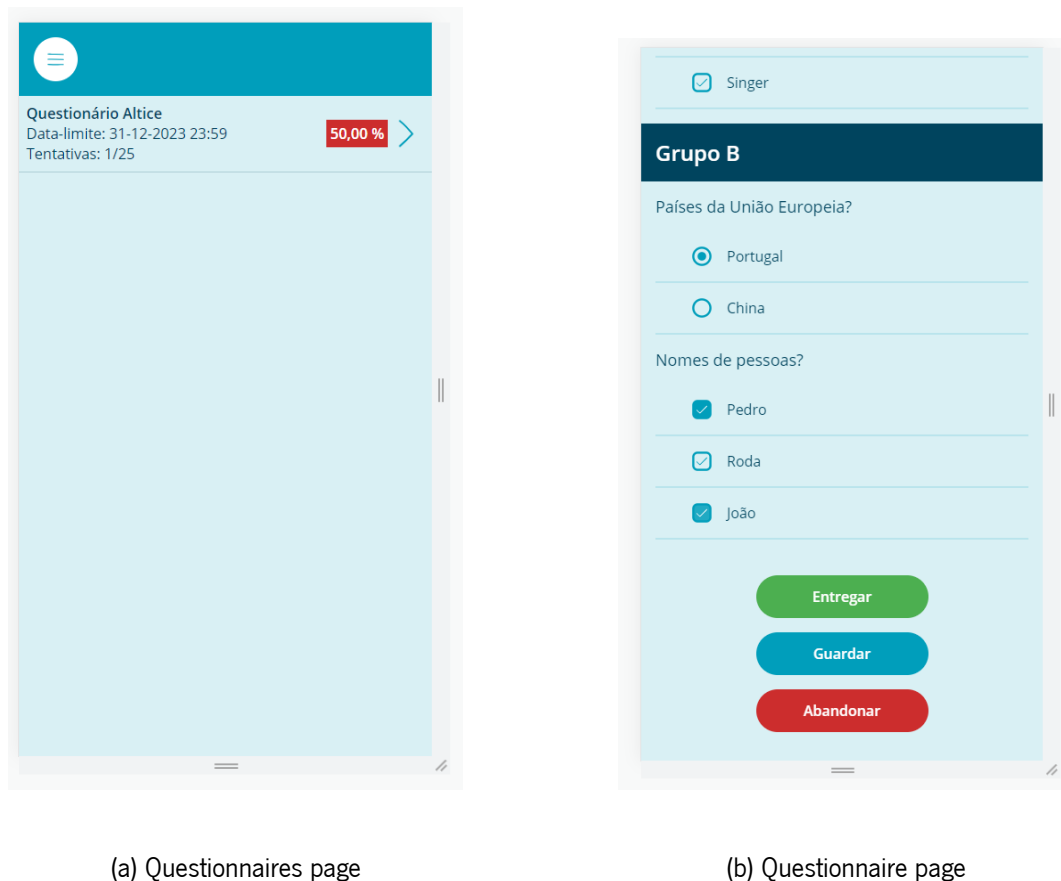


Figure 10: Questionnaire related pages

All software requirements regarding questionnaires were fulfilled apart from the ones shown in Table 32:

Number	Requirement
REQ-70	Upon submitting the questionnaire, the user should be redirected to a new page where they can view their submission and any corresponding feedback or grades
REQ-71	If a questionnaire has a time limit, a timer should be displayed during the questionnaire

Table 32: Unfulfilled questionnaires software requirements

Both these requirements are missing due to time constraints. REQ-70 is simple but not core to the platform, therefore other features were preferred over it. REQ-71 is an important feature but quite complex and time-consuming to develop therefore was also left out.

5.6 Architecture and Multi-language support

Other features that did not fit inside the core modules - authentication, user profile, notices, courses, contents and questionnaires - such as the architecture-related requirements and the multi-language support.

As for the architecture software requirements, all of them were successfully implemented, with special attention on **REQ-2 - The new client must preserve all existing functionalities from the original mobile website**. This requirement guarantees that long-time users will not be alienated when moving to the new interface.

About the multi-language support, once again all specified requirements were fulfilled. React and the styles library used, fit this requirement quite well and their use very much guaranteed that more than one language is supported and easily added.

With the new client-server architecture, other clients can be easily added in the future which was not possible before development. The work done during this project will have long-term benefits for the development team, who are now free to use more modern technologies for the application interface as well as an easier job maintaining and further developing these clients.

Finally, users now benefit from the multi-language support and new and improved visuals. The accessibility and usability were improved, as well as the range of devices supported due to the responsiveness of the new interface. The development team will also benefit from the new visuals because the styles library used has its dedicated development team, taking away some responsibility from the Formare developers and allowing them to focus on other aspects of the platform.

Chapter 6

Future Works and Conclusion

6.1 Future work

While the present development phase has provided a solid foundation, there remains room for further improvement and expansion. In this chapter, I will list what is missing from the current implementation, as well as what can be done to improve the platform.

6.1.1 Missing functionality

During the development of this project, I encountered several challenges that resulted in the inability to implement some functionalities. It is expected, during the development of any project, to encounter unforeseen problems that force a careful consideration of the project's priorities and necessities.

Whilst most of the key functionalities of the platform were successfully implemented, four important features are missing.

The first feature missing is the ability to display standardised content. While links, files and non-standardised contents are all currently available on the platform, contents that follow known standards, such as SCORM or AICC, are not. They were left out because they require a lot of work on the client side, meaning that not much code could be re-used. Most of the content types were already implemented and there were more important functionalities missing, therefore the decision was made to work on other, high-priority features.

The next two missing features were also left behind mostly due to time constraints; they are the questionnaires' timer and submission page. Regarding the former, not all questionnaires have time limits, however, for those who have, both the client and server should keep track of the time. As it stands, that is only being done on the server, therefore the user has no feedback on how much time left he has to submit his answers. As for the other questionnaire-related missing feature, after a user submits their answers,

they should be redirected to a page where they can view the result of their submission, as well as other important information defined by the questionnaire's configuration. Since at the moment no such page exists, the user is instead redirected to the page with the list of questionnaires, where they can only see their current mark. As for which of these should be a priority, I suggest the timer. While this one is far more complex to implement, it is also more important, as fair evaluation is a key aspect of any e-learning platform.

The final missing feature is complete error feedback on the client. Error messages are successfully being sent and translated to and by the client, however, aside from forms-related messages, they are not being shown. This is because error messages are supposed to be shown using an in-house toast-like component, that during this project's development has yet to be made available.

6.1.2 Design

Many possible improvements could be made to this thesis project, starting with the missing features. There is another, that I think is worth talking about related to the API and its design.

The web API created for this project was created for a very specific purpose, and that was to serve as a middleware between the client and the back office. That means that the API design is deeply connected to what the front end needs from the back office.

If efforts were made to redesign the API and rework the back office, these services could be used for more than just implementing Formare's front end. Perhaps they could be expanded upon to create a service that could supply different e-learning solutions with the needed backend logic, allowing clients to opt out of the interface to create their own e-learning experiences tuned in to their needs. There is potential for a new product to be born from these efforts.

6.1.3 Testing

While the current development phase did not encompass formal testing, it is vital to acknowledge the significance of testing as an essential aspect of any web platform's implementation process. After all, the incorporation of comprehensive testing methodologies becomes an imperative task for ensuring the robustness and reliability of the platform.

Should development resume in the future, dedicating effort to testing will be a vital step in validating and refining the web platform. By implementing various types of testing, such as unit testing, integration testing, functional testing, performance testing, and accessibility evaluations, we can systematically assess the platform's components and functionalities in isolation and as a cohesive system. The incorporation of

usability testing with real users will also be vital to obtain valuable feedback and improve the platform's user experience. In addition to promoting robustness and identifying bugs, testing will enhance the platform's scalability, stability, and security, thereby fortifying its readiness for deployment in real-world scenarios.

6.2 Conclusion

This past year I worked on this thesis with one final objective in mind:

- **Modernise Formare's user interface**

Through this project, I have discussed the evolution of distance learning, from letters to the internet and e-learning, and the role the latter plays in fostering accessible and dynamic learning experiences in both academic and professional environments.

Integrating a new user interface built with a modern web framework into an existing monolithic application like Formare's web forms presented a unique challenge: how to make the existing application logic available to be consumed by the new client. After carefully considering what options were available to make such a thing possible, it was decided to build a web API to mediate between the existing back office and the new client via HTTP, using a client-server architecture.

React's popularity as a choice for web platform development stems from its performance, scalability, and component-based architecture, all of which had some weight in adopting it to build Formare's new user interface. In parallel, the decision to utilise .NET and ASP.NET Core as the API framework was made to leverage the extensive capabilities and robustness offered by this framework. Furthermore, the existing back office was already built using .NET tools.

Despite choosing a back-end framework that used the same tools as the old back office to ease the integration process, the difference between versions introduced new challenges. Those added to the difficulties that had to be carefully explored to solve them, while maintaining the original behaviour of the application.

Looking at the future, the new user interface makes the platform easier and more satisfying to use, by different types of users. It also makes Formare easier to develop since the technologies are very popular and shared by other teams in Altice Labs, making it easier to find developers to further improve the application. Integrating the back office on a web API has also opened the door to creating a new service, one that could power other e-learning platforms and not just Formare.

In conclusion, the successful integration of React with the legacy web forms sets a precedent for enhancing existing monolithic applications, providing a compelling model for other platforms seeking to

modernise their interfaces and the tools supporting them.

Bibliography

- Hala Abushamleh and Shaidah Jusoh. Usability evaluation of distance education tools used in Jordanian universities. In *2021 Innovation and New Trends in Engineering, Science and Technology Education Conference (IETSEC)*, pages 1–5. IEEE, 2021. doi: 10.1109/IETSEC51476.2021.9440491.
- Beatrice Aguti, Gary B. Wills, and Robert J. Walters. An evaluation of the factors that impact the effectiveness of blended e-learning within universities. In *International Conference on Information Society (i-Society 2014)*, pages 117–121, 2014. doi: 10.1109/i-Society.2014.7009023.
- Shamim Akhter, Muhammad Kashan Javed, Syed Qasim Shah, and A Javaid. Highlighting the advantages and disadvantages of e-learning. *Psychol. Educ*, 58(5):1607–14, 2021.
- Amani Mubarak Al-Khatir Al-Arimi. Distance learning. *Procedia-Social and Behavioral Sciences*, 152: 82–88, 2014.
- Alaa Zuhir Al Rawashdeh, Enaam Youssef Mohammed, Asma Rebhi Al Arab, Mahmoud Alara, and Butheyna Al-Rawashdeh. Advantages and disadvantages of using e-learning in university education: Analyzing students' perspectives. *Electronic Journal of E-learning*, 19(3):107–117, 2021.
- Valentina Arkorful, Nelly Abaidoo, et al. The role of e-learning, advantages and disadvantages of its adoption in higher education. *International Journal of Instructional Technology and Distance Learning*, 12(1):29–42, 2015.
- Moncef Bari, Rachida Djouab, and Chi Phu Hoa. Elearning current situation and emerging challenges. *PEOPLE: International Journal of Social Sciences*, 4(2):97–109, 2018.
- Gary Berg and Michael Simonson. Distance learning. encyclopedia britannica, 2016. URL <https://www.britannica.com/topic/distance-learning>. Accessed on 23-10-2023.
- Blackboard. Blackboard learn help, 2023. URL <https://help.blackboard.com/Learn>. Accessed on 23-10-2023.

- Karolina Maria de Araújo Cordeiro. O impacto da pandemia na educação: A utilização da tecnologia como ferramenta de ensino. *Faculdades IDAAM*, 2020.
- Renée E Derouin, Barbara A Fritzsche, and Eduardo Salas. E-learning in organizations. *Journal of Management*, 2005.
- Formare. Homepage, 2020. URL <https://www.alticelabs.com/site/formare>. Accessed on 23-10-2023.
- Charlotte Nirmalani Gunawardena and Marina Stock Mclsaac. Distance education. In *Handbook of research on educational communications and technology*, pages 361–401. Citeseer, 2013.
- Nischal Guragain. E-learning benefits and applications. *Metropolia Ammattikorkeakoulu*, 2016.
- Mohd Hishamuddin Harun. Integrating e-learning into the workplace. *The Internet and higher education*, 4(3-4):301–310, 2001.
- Shan Jin. Design of an online learning platform with moodle. In *2012 7th International Conference on Computer Science & Education (ICCSE)*, pages 1710–1714. IEEE, 2012. doi: 10.1109/ICCSE.2012.6295395.
- Evelyn Kigozi Kahiigi, Love Ekenberg, Henrik Hansson, FF Tusubira Danielson, and Mats Danielson. Exploring the e-learning state of art. *Electronic Journal of e-learning*, 2008.
- Rachael Njeri Kibuku, Daniel Orwa Ochieng, and Agnes Nduku Wausi. e-learning challenges faced by universities in kenya: A literature review. *Electronic Journal of e-Learning*, 18(2):pp150–161, 2020.
- Altice Labs. History, 2023. URL <https://www.alticelabs.com/history>. Accessed on 23-10-2023.
- Maggie Lynch. E-learning during a global pandemic. *Asian Journal of Distance Education*, 15(1):189–195, 2020.
- Moodle. Accessibility - moodledocs, 2023. URL <https://docs.moodle.org/dev/Accessibility>. Accessed on 23-10-2023.
- Jacob Nielsen. 10 usability heuristics for user interface design, 2020. URL <https://www.nngroup.com/articles/ten-usability-heuristics>. Accessed on 23-10-2023.

- Mohammed Ouadoud, Mohamed Yassin Chkouri, Amel Nejjari, and Kamal Eddine El Kadiri. Studying and comparing the free e-learning platforms. In *2016 4th IEEE International Colloquium on Information Science and Technology (CiSt)*, pages 581–586. IEEE, 2016. doi: 10.1109/CIST.2016.7804953.
- Mike Paciello. *Web accessibility for people with disabilities*. Crc Press, 2000.
- Mirjana Radović-Marković et al. Advantages and disadvantages of e-learning in comparison to traditional forms of learning. *Annals of the University of Petroșani, Economics*, 10(2):289–298, 2010.
- Manijeh Sadeghi. A shift from classroom to distance learning: Advantages and limitations. *International Journal of Research in English Education*, 4(1):80–88, 2019.
- Sally Sambrook. E-learning in small organisations. *Education+ Training*, 45(8/9):506–516, 2003.
- Jeanne Schreurs and Ahmad Al-Huneidi. E-learning readiness in organizations. *International Journal of Advanced Corporate Learning (iJAC)*, 5(1):4–7, 2012.
- Statista. Percentage of global population accessing the internet from 2005 to 2021, by market maturity, 2022. URL <https://www.statista.com/statistics/209096/share-of-internet-users-in-the-total-world-population-since-2006>. Accessed on 23-10-2023.
- Maria Gil Lopes Maltez Suely Trevisan Araújo. Educação a distância: Retrospectiva histórica, 2010. URL <https://helena285.blogspot.com/2010/10/historia-da-ead.html>. Accessed on 23-10-2023.
- Alan W Tait. Education for development: From distance to open education. *Journal of Learning for Development*, 5(2), 2018.
- Maryam Tayebnik and Marlia Puteh. Blended learning or e-learning? *arXiv preprint arXiv:1306.4085*, 2013.
- WAI. Introduction to web accessibility, 2022. URL <https://www.w3.org/WAI/fundamentals/accessibility-intro>. Accessed on 23-10-2023.
- Ahmad Suffian Mohd Zahari, Suzila Mat Salleh, and Raja Mariam Raja Baniamin. Knowledge management and e-learning in organisations. In *Journal of Physics: Conference Series*, volume 1529, page 022051. IOP Publishing, 2020.

