esprit

Se former autrement

HONORIS UNITED UNIVERSITIES

2020 / 2021

Option : Software engineering

Internship thesis: Development of User Management, Continuous evaluation, Inscription form and Payment modules for an LMS Platform

Realized by: Houssem Bouzid

ESPRIT Supervisor: Miss Naouar Nessrine

Company Supervisor: Ben Abi Zid Mehdi

AUTHORIZATION TO FILE END OF STUDIES
INTERNSHIP REPORT

I authorize the student Houssem Bouzid to file his internship thesis

Academic supervisor: Naouar Nessrine
Date:
Signature

Company supervisor: **Ben Abi Zid Mehdi**

Date

Signature:

Acknowledgments

To my very dear mother

No words are strong enough to express my love and admiration

for her. Her understanding and tenderness make her a wonderful mother.

I thank her from the bottom of my heart and I dedicate this work to her, hoping that she will be proud of me. May God preserve her and grant her health and happiness.

To my dear father

Who is the best father in this world, for his encouragement, his trust, moral and material support and for his infinite love by expressing

my gratitude, my deep love and my passion.

Acknowledgments

We would like to express our gratitude to Miss Naouar Nessrine our supervisor within ESPRIT who agreed to monitor and evaluate this work. We would like to assure her of our deep respect and thank her for her participation at this particularly important moment for us. She gave us a lot of advice and advice during the project.

We would like to thank Slashup Studio and all the team who welcomed us to carry out our internship in good conditions.

We would also like to thank Ben Abi Zid Mehdi who supervised during this internship. The extensive knowledge he brought us throughout this project greatly contributed to its smooth running.

Finally, we warmly thank the members of the jury for their much-appreciated presence.

Table of contents

List of figures

List of tables

General introduction

Teaching is a practice, implemented by a teacher, aimed at transmitting skills to a pupil, a student or any other public within the framework of an institution educational. Many observers aware that the structures of traditional education have hardly changed since the beginning of the 19th century, despite the upheavals social and the evolution of the means of communication.

Traditional education is centered on lectures. Listeners are passive, often intellectually absent due to the difficult conditions imposed by the overload hourly. The main means used in this type of teaching: a lecture associated with directed and practical work, the “blackboard” and the “paper” support.

In addition, traditional education systems impose on all students a unity of place, unity of time, unity of action, unity of rhythm which implies a rigidity of the mechanisms and a difficulty of adequacy with the daily reality.

The trend towards improving the system from an educational point of view by resorting to conventional audiovisual means (slide shows, video sequences) has not resolved the problem. Indeed, the trainer must both present the course and undertake rather difficult technical manipulations.

Modern education requires means to facilitate student learning such as the choice of situation, the clarification of objectives and criteria, the choice of content, the setting implementing educational procedures, developing tools to help students build and appropriate knowledge and skills

Nowadays, a digital transformation has allowed the incorporation of new ecosystems, and one of them is e-learning. During the coronavirus pandemic, classrooms all over the world were forced to close and obliged 1.5 billion students and 63 million educators to switch to change face-to-face academic practices with digital tools wherever possible. This situation reveals the pros and cons of the educational system against the challenge of converting to distance learning. According to data provided by the World Bank, 83% of European citizens have access to the internet compared to 18% in emergent nations (LDC's) and 66% in Latin

America. Bates declared that the pandemic has shown the inequalities in the system and the need for universal and low-cost access to the internet for education and that the institutions must rethink their tuition policies to encourage students to move online by giving them advantages like discounts. From an economic point of view, this industry of online learning has grown in the last decades. According to Statista, this industry will be over 243 billion dollars in 2022.

the discourse around sustainability in education has developed in two ways:

- Education for sustainability that ensures environmental sustainability in education through educational practices
- Sustainability of education with e-learning that provides the possibility to change without outrunning the resource base and that can guarantee with the three pillars of digital learning: Resource management, Educational Attainment or continuous evaluation and Professional Development that implements new strategies the facilitate the adoption to change

Chapter 1 :General context

I- Introduction

During this chapter, we're going to present the general contest of the project and the company. In addition, we're going to present also the problematic, study of existence, our solution and finally our choice of development methodology

II- Project framework

We can define e-learning as an innovative web-based system on digital materials and forms to provide students a personalized and interactive learning experience. Also, we can be defined as a network to transfer skills, knowledge in sort of pieces of training through multiple devices such as tablets, computers, or cellular phones connected to the internet. This variety of devices makes it easy to learn anywhere, anytime. E-learning is used in national education programs, higher education programs, company training programs, and continuing education programs

We can also define it as "the use of new multimedia technologies of the Internet to improve the quality of learning by facilitating on the one hand access to resources and services, on the other hand, remote exchanges and collaboration ". It is part of information and communication technologies for education (ICTE) and allows for non-face-to-face activities. It is also most often the use of computers or mobile devices (smartphones, tablets, PDAs, etc.) connected to the Internet.

Distance training or education refers to educational situations and training systems for which the presence of the teacher or trainer is not necessary for the learning activity.

The development of distance learning responds to concerns and needs that are still very concrete:

III- Hosting organization:

Slashup Studio is a Startup specialized in venture building based in Tunisia and Paris. Its mission is help start-ups to lunch in market rapidly

Figure 1-1 slashup studio logo

IV- Study of the existing

The objective of this section is to study around the most popular E-Learning solutions known in the market. This study makes it possible to identify the strengths and weaknesses of each of these solutions. In what follows, we present an analysis of the existing, then we detail the critique of the existing.

1- Analysis of the existing

Continuing education is currently carried out in the traditional way: courses, students and trainers on site.

This type of training has many drawbacks such as:

-Constraint of the number of limited places

-Constraint of the reduced number of rooms

-High training load

In order to resolve these drawbacks, several tools have been created based on the newest technologies. Among which we can mention:

360Learning:

*Figure STYLEREF 1 \s 1 SEQ Figure * ARABIC \s 1 2: 360Learning logo*

360Learning platform for enterprises to form their employees and to evaluate their skills by giving them courses to study and projects to submit.

Teachable:

Figure 1-3 teachable logo

Teachable is an LMS or Learning management system. It is an online platform for creating and teaching courses. Content creators can create an online course and upload them on Teachable. It is like an online classroom

EDX:

Figure 1-4 edx logo

Edx is also an American MOOC provider founded by Harvard University and MIT. It hosts online university-level courses in multiple ranges of disciplines to the worldwide student including free courses. It also analyses how people use this platform to evaluate e-learning experience

Podia :

Figure 1-5 podia logo

Podia also a LMS platform for selling online courses by creating the storefront with no third-party plugins. It handles creating digital content and sending newsletters

2- Critique of the existing

As shown in the table below, these E-Learning solutions products and solutions offer various functionalities however, they have may drawbacks such as:

- The gap in the reviewing student Level quizzes and tests
- There is no platform that combines all features together

Tools	Advantages	Drawbacks
360Learning	Support paths Live webinars	Platform only for companies Doesn't have a free plan
Edx	Offers courses and paths given by famous universities Global recognition	Expensive cost per year ($33 368) Doesn't have webinar for teaching Platform only for

		universities Dose not accept payments in Tunisia
Teachable	Multilevel levels Admins Unlimited Videos Unlimited Live Classes Multiple payment methods	Doesn't have webinar for teaching Doesn't offer sessions Does not have free but it offers 15 days trial Doses not accept payments from Tunisia

When analyzing these platforms, we found that it has many gaps in reviewing student level by quizzes and tests and it lacks many features that can be added to make the student experience more efficient

V- Proposed solution: KOORS.io

The study of the existing made it possible to identify several anomalies that were detailed in the previous section. To make use of these anomalies we've prepared a design and implementation of a web portal bringing together all the e-learning functionalities.

In this solution we envision:

-An e-learning platform based on a web interface, which will avoid problems compatibility with the trainer's operating system and that of the students.
-A fast application (connection time, sharing time) and fluid (audio fluidity and video) in the Webinars Section. Given the number of important features, it should also offer ease of use and above all interface ergonomics.
-Brings together the functionalities of all the applications presented in the study of the existing.
-Multiple payments methods that cover all countries

VI- Choice of development methodology

1- Comparison between work methodologies

	Advantages	Drawbacks
Waterfall Model	-complete documentation for every step -easy to use -stable requirements	-large amount of documentation -not flexible -no ability to go back on steps
V-Model	-early place test placement -improved time management -immediate testing	-deficient flexibility for the model -insufficient risk analysis -no ability to dynamic changes

Agile	-division of the project on multiple iterations -flexible process to handle changes during the development of the product -minimized risks -consistent interaction with clients -high degree involvement for team member	-no define for project costs due the changes in development for the requirements - skilled team members are required -conflicts may come be with the new features
Spiral Model	-good documentation -early creation of the product -ability to add features late in the process -early prototype	-the need of specialist for risk analysis -success of the process depend on risk management -not suitable for small projects -rather expensive to use

2- Scrum methodology

Agile methodology represents the practice that supports continued iteration of product development and project tests during it lifecycle. It's considered as one of simplest but effective software development methodology and scrum in one of its frameworks
Agile methodology focuses on four main values:

1- Individual and interaction between the team over process
2- Software on a full documentation
3- Collaboration with the client rather the contractual negotiation
4- Flexibility to changes over following a blueprint or a plan

Also, there are agile 12 principles:

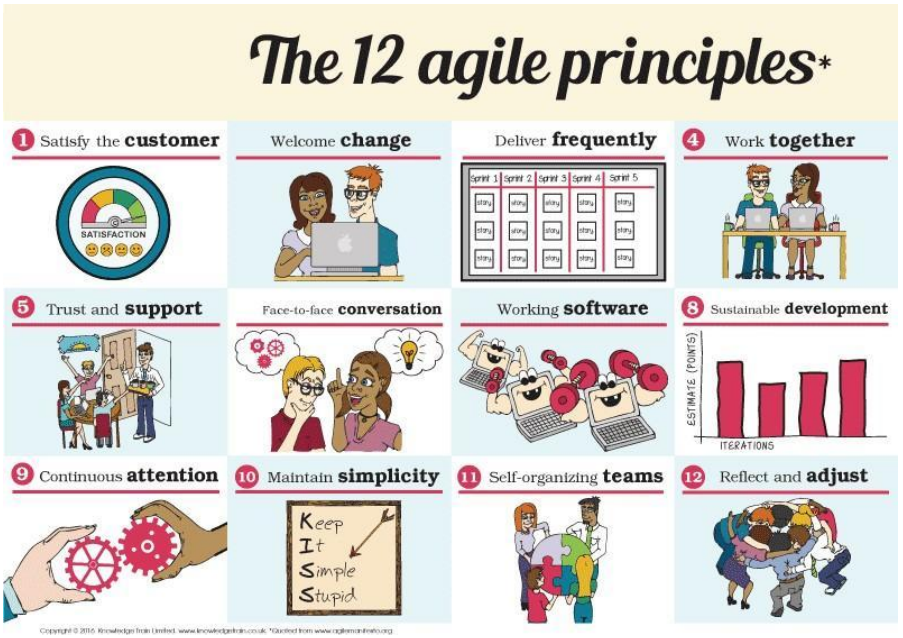


Figure 1-6: 12 agile principals

Scrum can be defined as an agile method or frameworks that focuses on managing tasks in team environment. It's based on three main roles:

Scrum Role	Mission
Product Owner	-He is the director of the product. -He communicates with the clients to get the product functionalities -
Scrum Master	-Team member with the main goal to optimize production capacities for the team -Helps the team in work independently with improving
Team	Operational team which ideally consists of less than ten people -The particularity of a Scrum team is that it is devoid of any internal hierarchy. -Self-organized

Table 1-1 SCRUM roles

There are also scrum events that has to be done on every iteration that lasts for 2 weeks to complete:

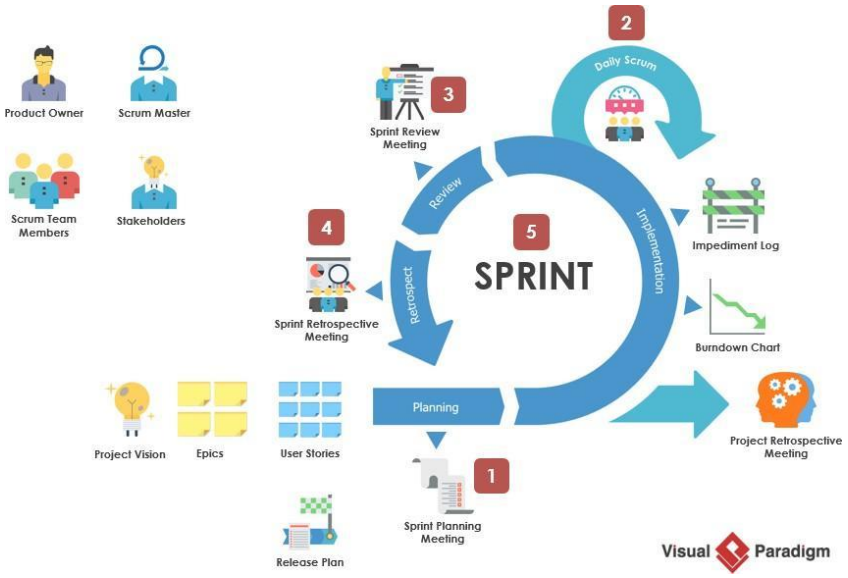


Figure 1-7 Sprint steps

Sprint planning: Happens in the beginning of the sprint where the team choose users stories and task they will work on during the sprint

Daily Scum: a 14-minutes meeting that synchronize activities between team members

Sprint Review: holds at the end of the sprint for the product adoption if needed

Sprint Retrospective: this is a event where the team inspect themselves in order to identify an improvement process to be implemented in the incoming sprints

Sprint: is defined as the length of the period to deliver sprint goals

VII- Conclusion

After conducting extensive research on the various existing methods, we chose the agile development method designed to shorten the software lifecycle by generating multiple versions. also, agile can be adopted to the team and company needs.

Chapter 2 : Preliminary Study

I- Introduction

During this chapter, we're going to introduce the main actors for our platform, functional and non-functional needs and finally used technologies to develop Koors

II- Framing of needs

1- Identifying actors

Our platform consists of creating an online school with their main actors and the right privilege and we can mention

Director: He is the creator of the school and he can manage school parameters and invite teacher and student to join

Teacher: the teacher can create a training and send a request to publish it to the director if the training is accepted

Student: he can join the school or get invited by the director and enroll in free or paid trainings

2- Functional needs

To develop Koors, we decided to develop these modules:

- User Management module: this module consists of managing different users in the platform and their info
- Inscription Request module: Some trainings need a specific skill to get accepted, that why the teacher creates an inscription request form to evaluate students before enrolling specific trainings
- Continuous Evaluation Module: During the trainings or at the end of it, the student must answer to several quiz's and in the end, he will pass a final exam to evaluate his skills
- Payment modules: not all the trainings are free, that's why we developed a payment module to give the student an opportunity to pay for trainings

3- Non functional needs

Non functional needs permits to the platform to works correctly after developing the functional needs and we can mention:

- Security: focuses all assets to be protected throughout the lifecycle
- Developability: refers to adapting systems and the experience in updating and changing the existence code
- Testability: includes to need to be able to validate requirements of the system

III- Needs modeling

1- Global use Case

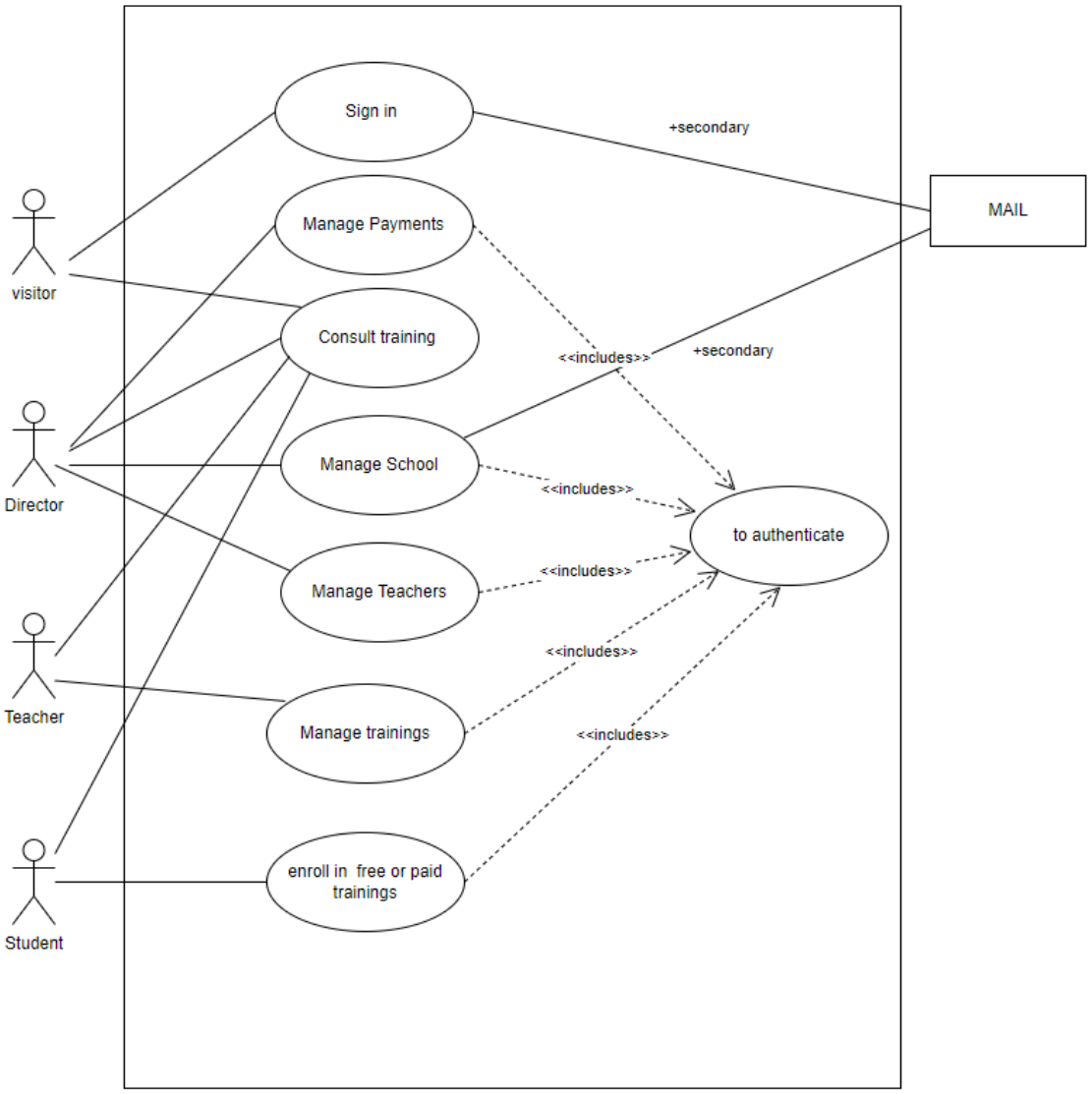


Figure 2-1 General use case

IV- Product Backlog

	Feature	ID	User Story	ID Task	Task	Estimation
1	User Management Module	1.1	Setting User Management database architecture	1.1.1	Creating entities of the module	2
				1.1.2	Creating development environments	1
		1.2	As a visitor I can create account	1.2.1	Account creation (back)	3
				1.2.2	Account creation (front)	3
				1.2.3	behavioral testing / Unit testing	2
				1.2.4	Documentation	1
				1.2.5	Api Security	1
		1.3	As user, I can log in	1.3.1	Log in (back)	2
				1.3.2	Log in (front)	2
				1.3.3	Behavioral testing (cucumber)	1
				1.3.4	Unit testing (Junit)	1
				1.3.5	Api Security	1
		1.4	As connected User , I can preview my profile	1.4.1	Preview profile (back)	2
				1.4.2	Preview Profile (front)	3
				1.4.3	Unit and Behavioral testing	2
				1.4.4	Documentation	1
				1.4.5	API Security	1
		1.5	As connected user, I can modify my profile information's	1.5.1	Modify profile (back)	2
				1.5.2	Modify profile (front)	3
				1.5.3	Unit and Behavioral testing	2
				1.5.4	Documentation	1
				1.5.5	Securing the api	1
		1.6	As connected, user I can delete my profile	1.6.1	Delete account (front)	1
				1.6.2	Delete account (back)	1
				1.6.3	Unit testing	1
				1.6.4	Behavioral testing	1
				1.6.5	Securing the api	1
		1.7	As a director, I can invite a professor	1.7.1	Invite professor (back)	2
				1.7.2	Invite professor (front)	1
				1.7.3	Unit testing	1
				1.7.4	Behavioral testing	1
				1.7.5	Securing testing	1
2	Inscription request Module	2.1	As professor , I can create Course inscription form	2.1.1	Form creation (backend)	3
				2.1.2	Form creation (frontend)	3
				2.1.3	Unit / behavioral testing	2
				2.1.4	Api security	1
				2.1.5	documentation	1
		2.2	As student , I can apply to a course with request inscription form	2.2.1	Submit a request to take a course (backend)	2
				2.2.2	Submit a request to take course (frontend)	3
				2.2.3	Unit / behavioral testing	2
				2.2.4	Api security	1
				2.2.5	Api documentation	1
		2.3	As professor, I can preview inscription requests	2.3.1	Display students' requests(backend)	3
				2.3.2	Display students' requests (frontend)	3
				2.3.3	Unit / behavioral testing	2

				2.3.4	Api Security	1
				2.3.5	Api documentation	1
		2.4	As professor , I can accept or decline requests	2.4.1	Accept/refuse requests (backend)	2
				2.4.2	Accept/refuse requests (frontend)	2
				2.4.3	Unit / behavioral testing	1
				2.4.4	Api security	1
				2.4.5	Api documentation	1
3	Continuous evaluation Module	3.1	As professor, I can create an exam	3.1.1	Exam creation (back)	5
				3.1.2	Exam creation (front)	3
				3.1.3	Api security	1
				3.1.4	Unit / behavioral Testing	2
				3.1.5	Api documentation	1
		3.2	As professor , I can create questions	3.2.1	Open question creation (backend)	3
				3.2.2	Open question (frontend)	2
				3.2.3	Api Security	1
				3.2.4	Unit / behavioral testing	2
				3.2.5	Api documentation	1
		3.3	As a student I can pass quiz’s	3.3.1	Pass quiz (back)	5
				3.3.2	Pass quiz (front)	3
				3.3.3	Unit / behavioral testing	2
				3.3.4	Api security	1
				3.3.5	Api documentation	1
		3.4	As a student I can pass a final exam	3.4.1	Pass final exam (backend)	3
				3.4.2	Pass final exam (front)	2
				3.4.3	Unit / behavioral testing	2
				3.4.4	Api security	1
				3.4.5	Api documentation	1
		3.5	As teacher, I can preview final exam submissions	3.5.1	Preview list of exams (backend)	5
				3.5.2	Preview list of exams (frontend)	3
				3.5.3	Preview a specific exam (frontend)	3
				3.5.4	Unit / behavioral testing	1
				3.5.5	Api security	1
				3.5.6	Api documentation	1
		3.6	As teacher, I can set marks to open questions	3.6.1	Set mark for a open question (backend)	2
				3.6.2	Set mark for open question (front	3
				3.6.3	Unit / behavioral testing	2
				3.6.4	Api security	1
				3.6.5	Api documentation	1
		3.7	As student I can preview my exam or quiz submission and mark	3.7.1	Get a specific exam details (backend)	3
				3.7.2	Display test result	3
				3.7.3	Unit / behavioral testing	2
				3.7.4	Api security	1
				3.7.5	Api documentation	1
4	Payment Module	4.1	As director, I can fix course, session path price	4.1.1	Fix content price (back)	5
				4.1.2	Fix content price (front)	3
				4.1.3	Unit / behavioral testing	2
				4.1.4	API security	1
				4.1.5	API Documentation	1
		4.2	As director, I can manage courses price	4.2.1	Manage content price (back)	5
				4.2.2	Manage content price (front)	3
				4.2.3	Unit / behavioral testing	2

				4.2.4	Securing the API	1
				4.2.5	API documentation	1
		4.3	As director, I can set payment method for his school	4.3.1	Choose payment method (back)	3
				4.3.2	Choose payment method (front)	3
				4.3.3	Unit / behavioral testing	2
				4.3.4	API Documentation	1
				4.3.5	Securing API	1
		4.4	As director, I can consult payment history	4.4.1	Consult payment history (back)	5
				4.4.2	Consult payment history (front)	3
				4.4.3	Unit / Behavioral testing	2
				4.4.4	Securing API	1
				4.4.5	API documentation	1
		4.5	As a student, I can pay a training with paymee	4.5.1	Paymee payment(Back)	5
				4.5.2	Paymee payment(Front)	3
				4.5.3	Unit / Behavioral testing	1
				4.5.4	Securing API	1
				4.5.5	API documentation	1

Table 2-1 Backlog

V- Release planification

After creating the product backlog with all user stories, tasks, sub-tasks and tests tickets, we have organized planification meeting to divide work into releases and in the end we decided to go with:

- Release 1: User Management Module

- Release 2: Inscription request Module and Continuous Evaluation Module

- Release 3: Payment module

VI- Work environment and technical choices

1- Work environment

Visual studio code:

Figure 2-2 visual studio code logo

Vs code is an open open-source and cross-platform editor represented by Microsoft and offers many features such as debugging, code refactoring and can integrate many extensions from it marketplace

IntelliJ idea community:

Figure 2-3 Intellij idea logo

IntelliJ idea community is an IDE developed by JetBrains for java programmers and considered as one top-choices IDE for java developers

Postman:

Figure 2-4 Postman logo

Postman is a JavaScript application for testing developed APIs that consumes web services to understand received data structure and values

Figma:

Figure 2-5 figma logo

Figma is a victor graphic editor web-based tool with additional features on desktop version

Draw io:

Figure 2-6 Draw io logo

Draw Io is an online open-source software for UML modeling, making flowcharts and process diagrams

Jira:

Figure 2-7 jira logo

Jira is project management software developed by Atlassian used to ship product faster that adapts the agile methodology for product management

2- Technical choices

Java:

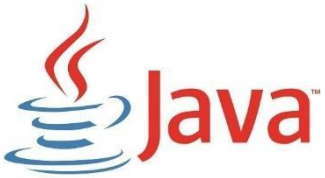


Figure 2-8 java logo

Java is one more common programming language released by Sun Microsystems. It's a high-level and object-oriented language and it's designed to let developers to write ones, write anywhere (WORA) and we are using Java8 because it is long support version (until 2030)

JavaScript:

Figure 2-9 JavaScript logo

JavaScript is a interpreted, lightweight, just-in-time programming time and it is the most well-known as scripting language for web pages. Also, it's a multi-paradigm, dynamic, single-threaded language

TypeScript:

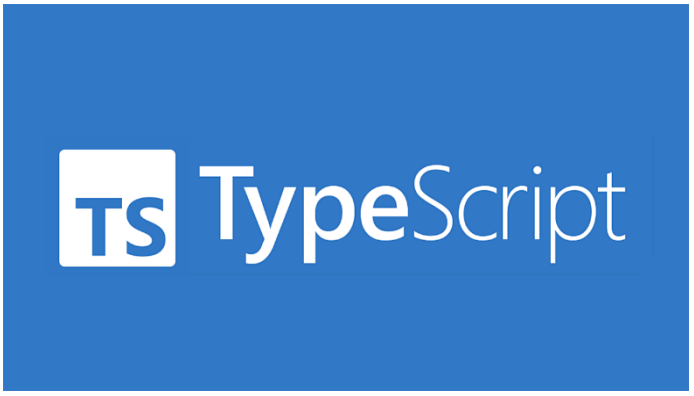


Figure 2-10 typeScript logo

TypeScript is a strict syntactical programming language developed by Microsoft and considered as a superset of JavaScript. It's designed for the development of large application

SpringBoot:

Figure 2-11 Spring boot logo

Spring Boot represents open java solution for developing back-end application developed by Pivotal. It offers a pre configurable project application after it creation. With it , programmers can start quickly without losing time on preparing a spring application

JavaScript Object Notation (JSON)

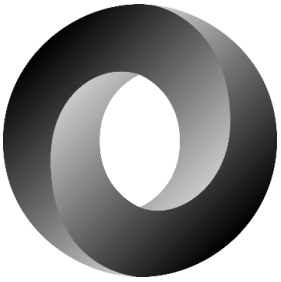


Figure 2-12 JSON logo

JSON represents textual data structure from the object notation of JavaScript programming language. It allows to represent structured data like XML

NodeJs:

Figure 2-13 NodeJs logo

NodeJS represents an open source, backend JavaScript and cross platform tools running using V8 engine that executes outside web browsers

Json Web Token:

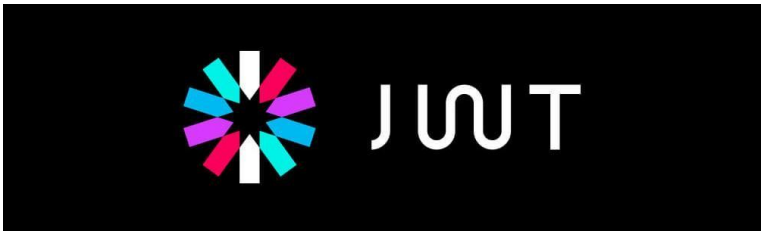


Figure 2-14 JWT logo

JWT is a standard that allows the exchange of secured tokens between different parts that ensure the integrity in the digital signature

PostgreSQL:

Figure 2-15PostgresSQL

PostgreSQL represent open-source relational database management system developed at Berkey, California University as an enterprise-class database, it offers many advantages save points (nested transaction) , Multi-Version-Concurrency Control and online backups

Swagger:

Figure 2-16 Swagger logo

Swagger is a library that allows you to describe APIS, helps the design, build REST API and document it for other developers

Cucumber:

Figure 2-17 Cucumber logo

Cucumber is an open-source testing tool used for Behavior Driven Development (BDD) testing in Java and node

Junit:

Figure 2-18 JUnit logo

Junit is an open-source testing framework to execute unit and integration testing for java programming language

VII- Application architecture

1- Physical architecture

Koors infrastructure is an AWS Kubernetes multitenancy architecture. A multitenant architecture is an architecture with the essential goal which is serving as many users as it can at the same time with processing capacity that grows with the growth of the users' number. it also insures the maintaining of stability, performance and durability.
In addition, it enables data isolation of each user and concurrent processing.

Similarly, it ensures that each user have ample access to resources when it comes with resources sharing.

Multi-tenant Kubernetes workload architecture:

To have multi-tenant architecture, we have need EKS clusters on Amazon with multiple tenants in workloads. In the diagram below, we have isolated namespaces with multiple tenants.

> A namespace is a logical presentation that divides cluster resources that joins multiple reassures
> On the EKS clusters, we have isolated and independent components of the applications. these isolated components can be storage, computer resources… and we can ensure the isolation using various techniques such as network policies

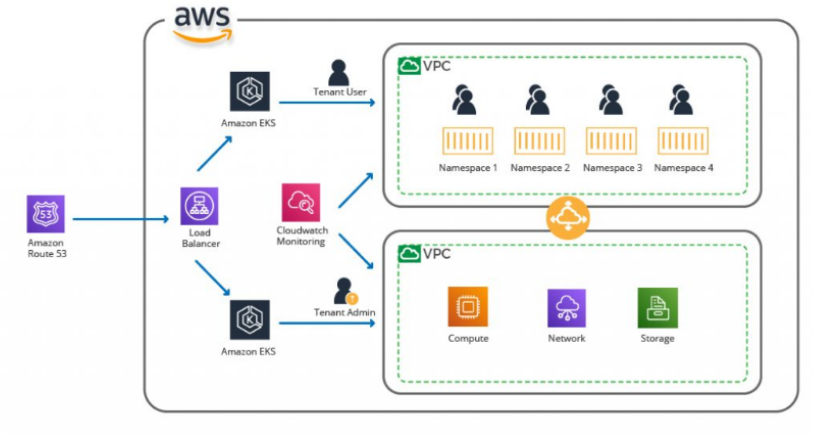


Figure 2-19 Multi-tenant Kubernetes workload architecture

EKS and isolation layers: there are many layers that helps with isolation for a Kubernetes application:
Cluster: is a collection of nodes and represents the management layer for containers that ensures extreme isolation in network
Node : the node is a physical or virtual machine that contains multiple pods that helps hardware or hypervisor for the resources isolation
Pods: pods are groups of containers that isolate networks and helps in micro-segmentation of containers
Containers: it represents the main layer of isolation; however, it can't isolate the network and identity but it isolates from noisy neighbors.

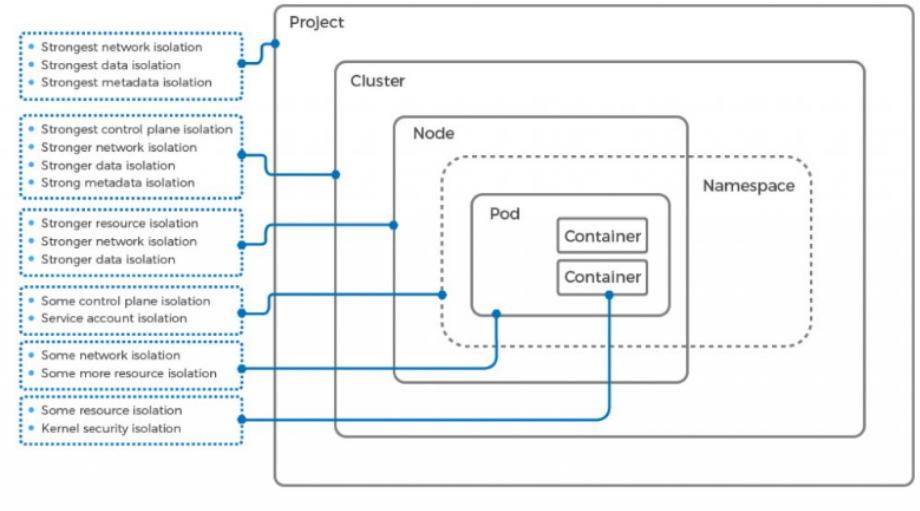


Figure 2-20 EKS and isolation layers

master node: with is triggered after a push into one of koors GitLab repositories, the deploy jobs started, then the new version of code will be pushed into the microservices replicas witch are founded in pods.
inside the pods we can find secrets keys such as Kong secret key , database passwords, and api's keys
welcome node: after the code is deployed, the code will be be used in welcome nodes that will be used when we call the microservice endpoints

2- Logical architecture

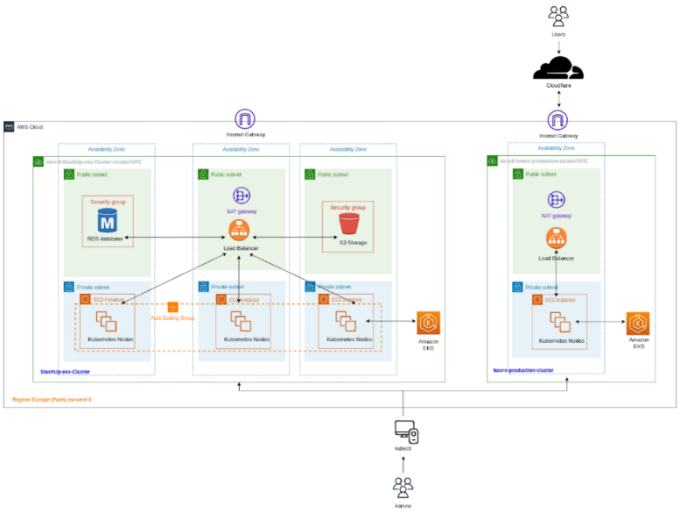


Figure 2-22 KOORS logical architecture

As shown in the diagram, Koors infrastructure is divided on two main sections:

- Production pod: Koors production pod which is isolated from the other environments to ensure isolation from incidents
- Test, Developments and pre-production environment: it's a pod that contains theses three environments and it's serves on testing the platform before deploying new code and functionalities into the production environment

Backend Architecture:

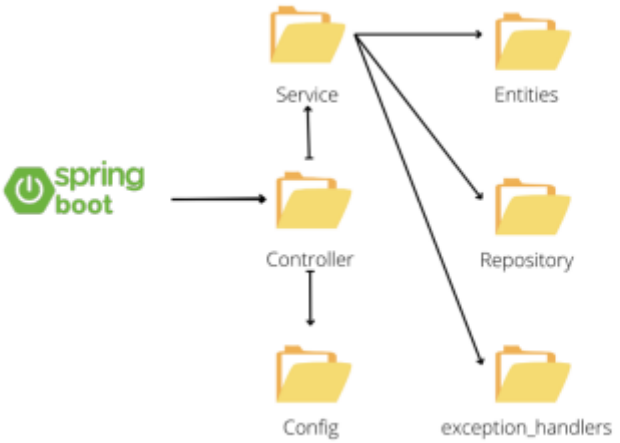


Figure 2-23 SPRING boot architecture

Frontend Architecture:

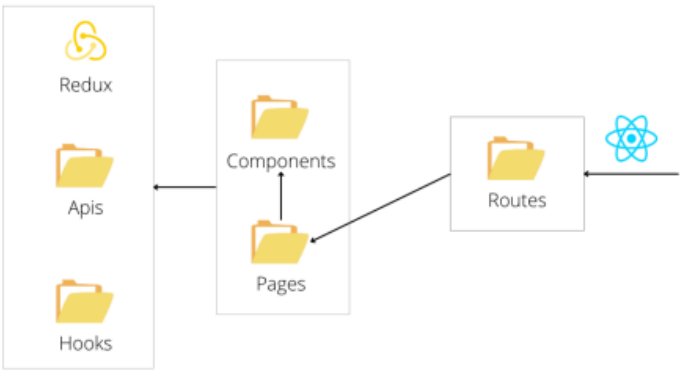


Figure 2-24 react architecture

Routes represents the entry point which are used to used to display platform pages developed of one or several components to display data consumed by endpoints or from redux state

3- Microservice Architecture

We can define microservice architecture as a method used in software development that breaks down an application in order to isolate key functions called as "services" designed to meet a unique need such as payment or chat. It gives to every service the advantage of being modular and independent that allows to be developed without affecting other services and finally being deployed. This type of architecture build represents the opposite of monolithic architecture with represents building the application as a single project or unit

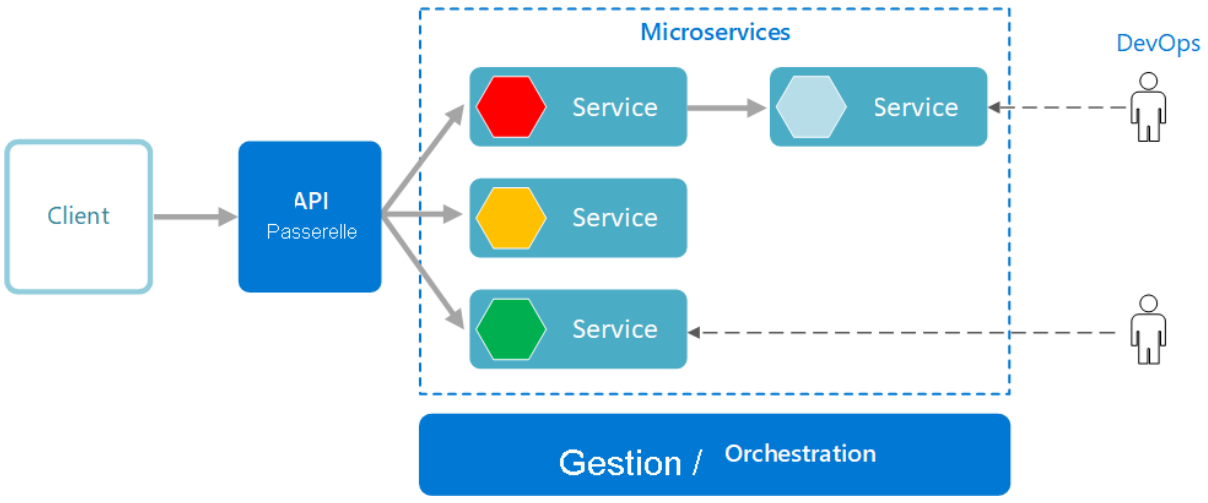


Figure 2-25 Microservice architecture

Advantages of microservice:

1- Independent Development: developers can choose the right technologies for each microservice and not restricted by the choices made at the outset of the project
2- Independent deployment: Microservices can be deployed independently and it can be deployed without the need to redeploy the hole application and that makes the fix of bugs or adding new features easier. In addition, it can run builds, test after each development and finally deployment
3- Independent scaling: Microservice architecture can scale independently to satisfy it needs, optimizes time and cost because there is no need to scale the hole application
4- Targeting small teams: developers can target only one microservice, facilitating the understanding of the code and the integration of new members within the team
5- Small code: Adding new functionalities requires changing the code in many places but in microservice architecture, we can minimize dependencies and it makes easier when adding new code or futures
6- Data isolation: each microservice have it own private database and it helps in not affecting the other services in case of updating or deleting schema

7- Resilience: Thanks to microservice architecture, critical fault points are significantly reduced. When a microservice goes down, the entire application dose not stop running as in the case of monolithic architecture. In addition, we can isolate errors and tis make it easier to correct

Orchestration:

When using microservice architecture, there is no need for services to communicate with each other, that's why we use orchestration which is a simple approach the be set up in since we have dependencies between microservices that we want to eliminated or reduced and this is an example of a microservice with orchestrator:

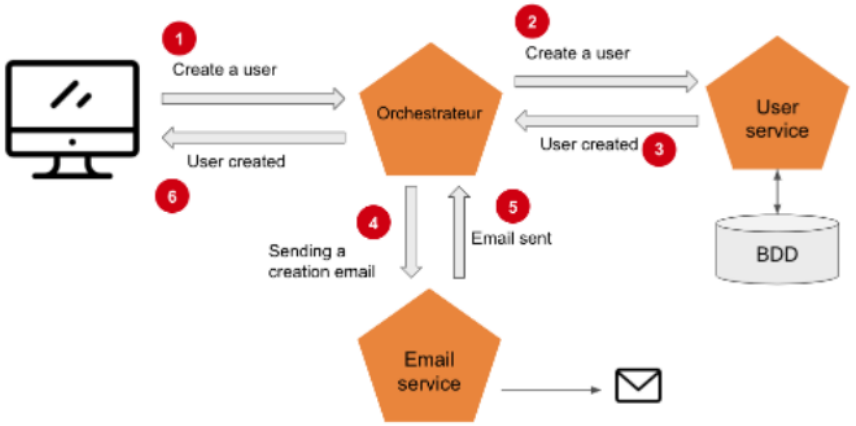


Figure 2-26 Microservice with orchestration example

4- MVC

a- What is MVC

MVC represents a design pattern that assists in building and constructing application frameworks. It consists of three sections named Models, Controllers and Views. It is a trendy design used by all types of programming languages like JavaScript, java, PHP….

b- Different Sections of MVC

The Model Section:

This section corresponds to the data and logic part in the application. It may have many connections to one or many other models in the applications and it is there where the data can be manipulated, added and stored. When the data changes, the model will typically alert the views and when the logic changes, the controller will be alerted to update the view if it's needed and we can be represented as the cooks if we think that MVC can be represents a restaurant

The Controller Section:

This section represents the intermediate for the model and view parts. It transmits the data from and to application by transferring, translating the data from the view and interpreting the user in order to connect the model and view

The View Section:

This section represents the area of the application where the user has the possibility to directly interacts. It's linked to the model and displays the data from it

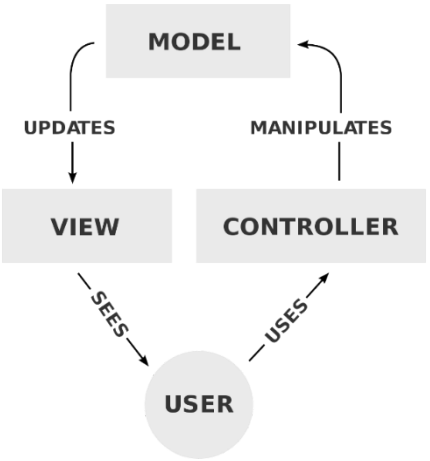


Figure 2-27 MVC

c- Why MVC?

MVC can give many benefits in software development and we can mention:

- Simultaneous Development: the project can be developed by many developers in the same time
- High cohesion: it enables the grouping of the logical parts together
- Low coupling: the MVC architecture gives us the possibility to have a simple and stable interface isolated from the internal implementation

- Ease of modification: MVC gives us the possibility to easily update and add code to the application while the development phase or the bugs fixing

VIII- Conclusion

In this chapter, we have specified our functionalities with mentioning tasks to develops, technologies to use and architecture we need in order to develop our platform

Chapter 3 : Release 1: User Management Module

I- Introduction

For the user management micro-service, we have divided users in two types in order to create schools and manage users in our platform:

-Natural Person: Represents the ordinary users of the platform such as professors, students…

-Legal Person: Represents schools in our case

II- Release Backlog

	Feature	ID	User Story	ID Task	Task	Estimation
1	User Management Module	1.1	Setting User Management database architecture	1.1.1	Creating entities of the module	2
				1.1.2	Creating development environments	1
		1.2	As a visitor I can create account	1.2.1	Account creation (back)	3
				1.2.2	Account creation (front)	3
				1.2.3	behavioral testing / Unit testing	2
				1.2.4	Documentation	1
				1.2.5	Api Security	1
		1.3	As user, I can log in	1.3.1	Log in (back)	2
				1.3.2	Log in (front)	2
				1.3.3	Behavioral testing (cucumber)	1
				1.3.4	Unit testing (Junit)	1
				1.3.5	Api Security	1
		1.4	As connected User , I can preview my profile	1.4.1	Preview profile (back)	2
				1.4.2	Preview Profile (front)	3
				1.4.3	Unit and Behavioral testing	2
				1.4.4	Documentation	1
				1.4.5	API Security	1
		1.5	As connected user, I can modify my profile information’s	1.5.1	Modify profile (back)	2
				1.5.2	Modify profile (front)	3
				1.5.3$l	Unit and Behavioral testing	2
				1.5.4	Documentation	1
				1.5.5	Securing the api	1
		1.6	As connected, user I can delete my profile	1.6.1	Delete account (front)	1
				1.6.2	Delete account (back)	1
				1.6.3	Unit testing	1
				1.6.4	Behavioral testing	1
				1.6.5	Securing the api	1
		1.7	As a director, I can invite a professor	1.7.1	Invite professor (back)	2
				1.7.2	Invite professor (front)	1
				1.7.3	Unit testing	1
				1.7.4	Behavioral testing	1
				1.7.5	Securing testing	1

Table 3-1 Release 1 backlog

III- Modelling

1- Use case Diagram

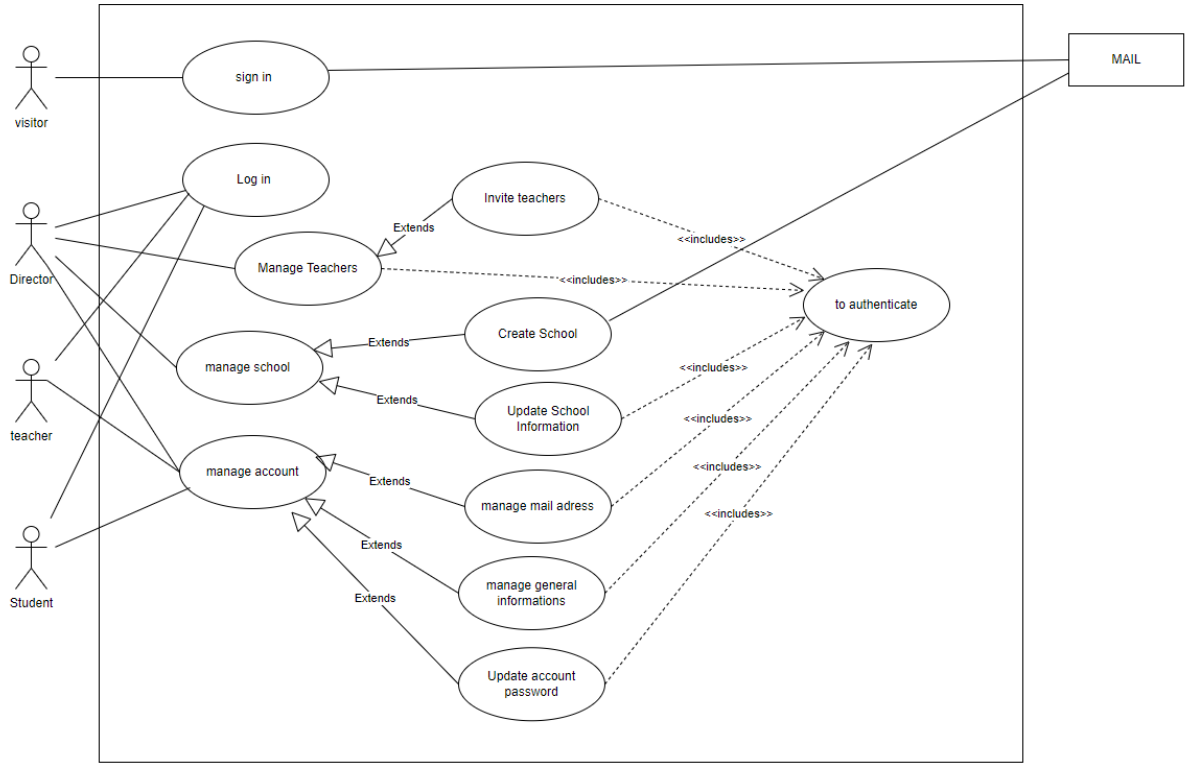


Figure 3-1 User management use case

2- Class Diagram

Figure 3-2 User management class diagram

IV- Release deliverable

1- Sign In:

In order to access to the platform, the user must sign in to the platform as director, teacher or student. After choosing the type of inscription, an email is sent in order to verify the mail address . Next , he complete the sign in by typing general information and uploading a profile photo

Use Case	Create an account
Actors	Visitor
Hypothesis	The visitor can't sign in

Pre-conditions	any visitor can create an account
Initialization	This use case starts when the visitor decides to create an account
Description of the scenario	- The user clicks on create account - The user gives his mail address and password - The user activates his account by clicking on the link sent to this mailbox - The user complete joining to the platform by completing his general information and uploading his profile photo if he wants
Post-conditions	The user will be added to the database

Table 3-2 sign in textual use case

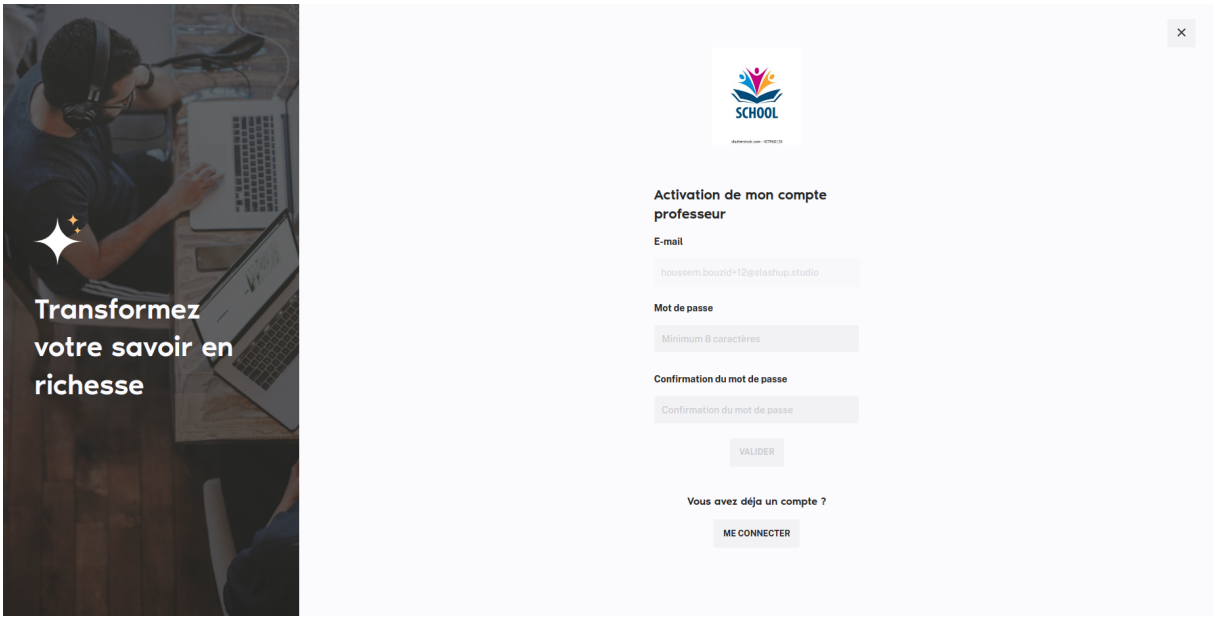


Figure 3-3 Create professor account step 1

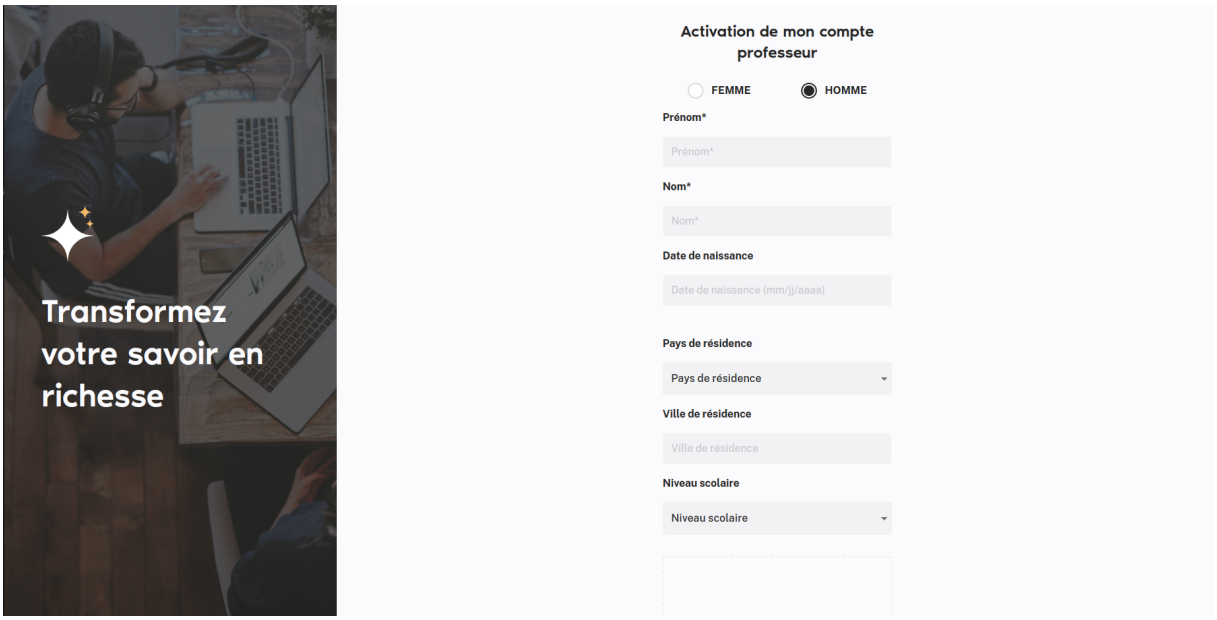


Figure 3-4 create professor account step 2

In order to complete the sign in process, a verification mail is sent to verify that the mail.

2- Managing user account

Users can manage their profile information's like the address, city and profile picture

Use Case	Manage user account
Actors	Any user
Hypothesis	This user wants to modify
Pre-conditions	any visitor can create a school after choosing the suitable plan to him
Initialization	This use case starts when the visitor decides to create a school
Description of the scenario	- The admin sees the request of creating a school - The admin creates the school after completing it general information - The director activates his school by clicking on the link sent to this mailbox - The director compete the school profile
Post-conditions	The school we be added to the database and it generated sub domain

Table 3-3 Update user account info

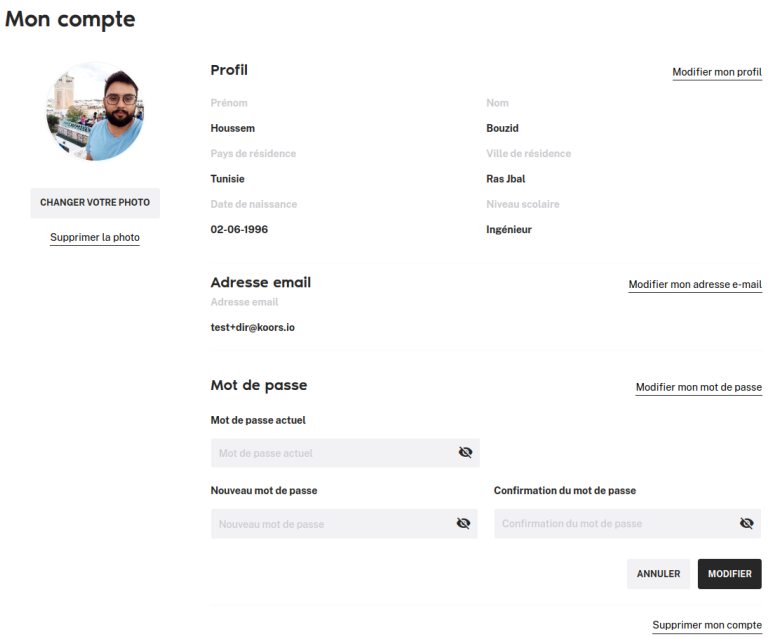


Figure 3-5 Update user account password

And for the Security of the account, the users can change their password or mail address by confirming the changes by clicking the link sent in a mail to make sure that the users themselves whom changes these information

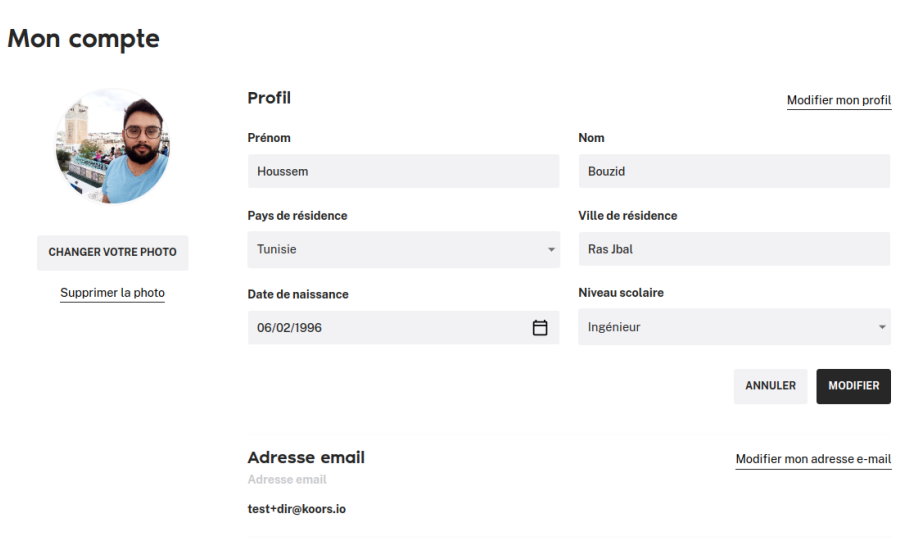


Figure 3-6 Update user general information's

3- Creating a new School

Any user can create a school and became it director by choosing the plan that suits him and sending info's about his school the admin.

A mail with confirmation we be received that includes the sub domain from Koors to access into his school, after that he must complete the school info's, upload it logo and the cover photo

Use Case	Create a school
Actors	Visitor
Hypothesis	This user wants to create a school
Pre-conditions	any visitor can create a school after choosing the suitable plan to him
Initialization	This use case starts when the visitor decides to create a school
Description of the scenario	- The admin sees the request of creating a school - The admin creates the school after completing it general information - The director activates his school by clicking on the link sent to this mailbox - The director compete the school profile
Post-conditions	The school we be added to the database and it generated sub domain

Table 3-4 Create School textual use case

4- Inviting teachers to join to the school

In Order the build the school team, the director can invite professors to join his school by send them mails. Next, the professor can access to the school by clicking on link sent

Use Case	Inviting a new teacher to the school
Actors	director
Hypothesis	A director wants to add a new teacher
Pre-conditions	During setting the list of teachers, the director wants to add a new teacher
Initialization	This use case starts when the director clicks on add professor button
Description of the scenario	- The director clicks on add teacher button - The director writes the teacher mail address - The director clicks on invite button
Post-conditions	After clicking on invite button, an invitation mail will be send to the teacher

Table 3-5 Invite teacher to school use case

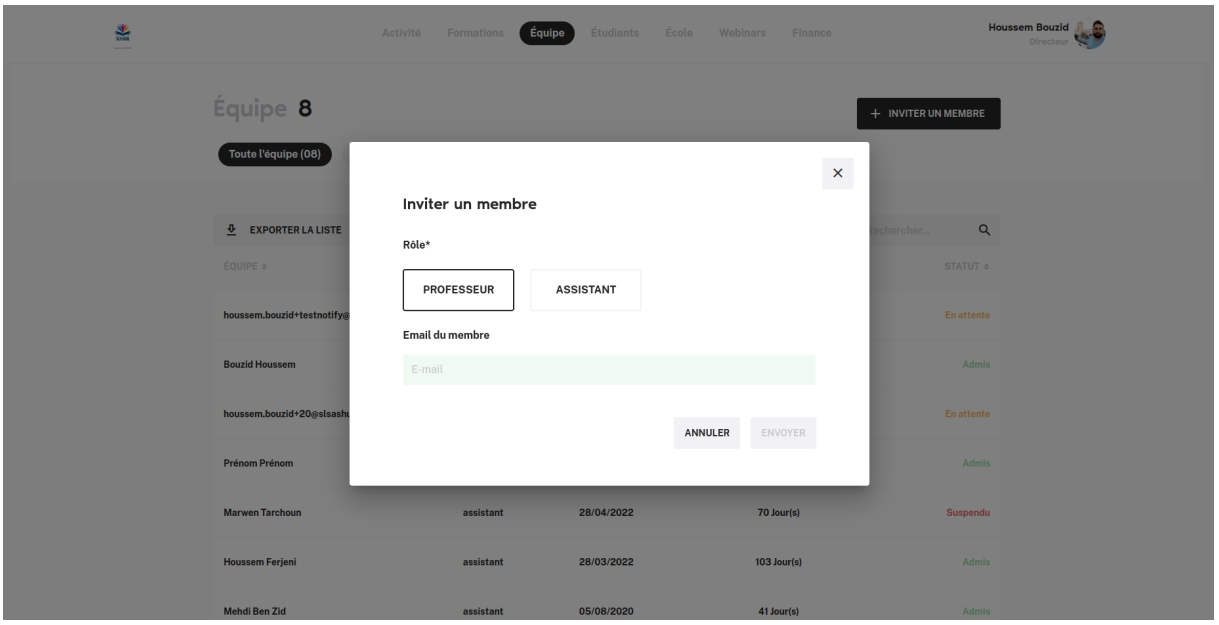


Figure 3-7 Invite professor to join school step 1

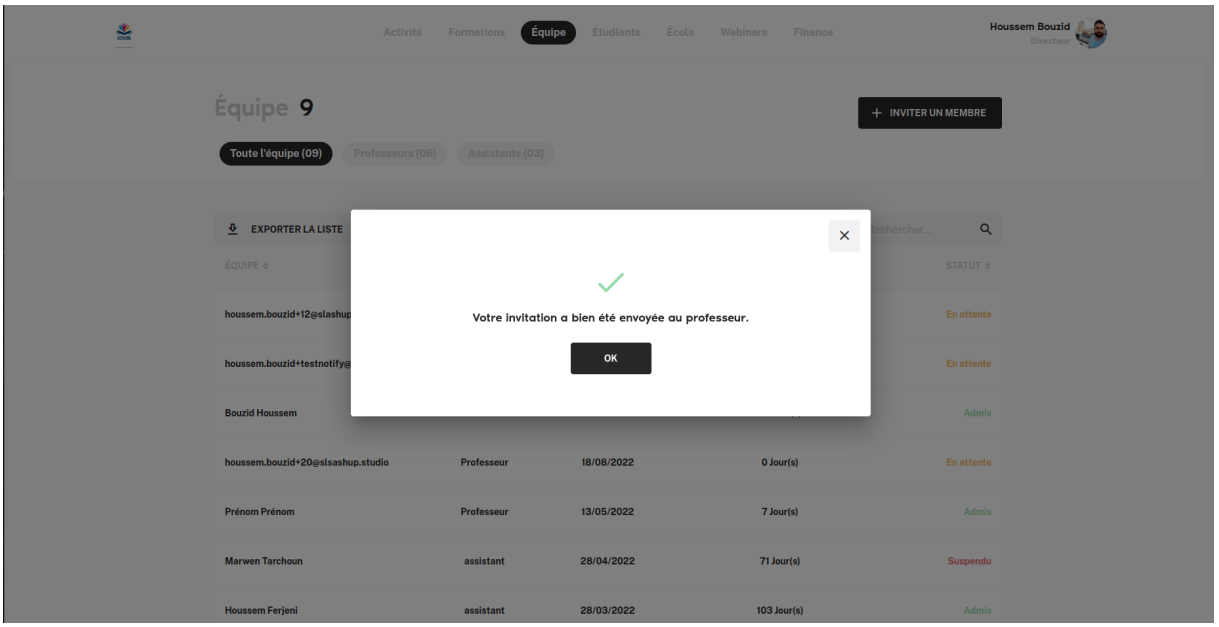


Figure 3-8 Invite professor to join school step 2

An invitation mail will be received in teacher mailbox that will redirect him to sign in form in order to create account and to log in as professor in the end in the director school

V- Conclusion

In this release, we have developed the frontend and the backend part of the user management module, also we have corrected declared bugs

Chapter 4 : Release 2: Inscription request module and continuous evaluation module

I- Introduction

This Release consists of two parts:

- Inscriptions request module: represents an entry used by the teacher in order to evaluate the student skills before taking the training
- Continuous evaluation module: represents the evaluation during and in the end of training

II- Release Backlog

	Feature	ID	User Story	ID TASK	Task	Estimation
2	Inscription request Module	2.1	As professor , I can create Course inscription form	2.1.1	Form creation (backend)	3
				2.1.2	Form creation (frontend)	3
				2.1.3	Unit / behavioral testing	2
				2.1.4	Api security	1
				2.1.5	documentation	1
		2.2	As student , I can apply to a course w ith request inscription form	2.2.1	Submit a request to take a course (backend)	2
				2.2.2	Submit a request to take course (frontend)	3
				2.2.3	Unit / behavioral testing	2
				2.2.4	Api security	1
				2.2.5	Api documentation	1
		2.3	As professor, I can preview inscription requests	2.3.1	Display students' requests(backend)	3
				2.3.2	Display students' requests (frontend)	3
				2.3.3	Unit / behavioral testing	2
				2.3.4	Api Security	1
				2.3.5	Api documentation	1

		2.4	As professor , I can accept or decline requests	2.4.1	Accept/refuse requests (backend)	2
				2.4.2	Accept/refuse requests (frontend)	2
				2.4.3	Unit / behavioral testing	1
				2.4.4	Api security	1
				2.4.5	Api documentation	1
3	Continuous evaluation Module	3.1	As professor, I can create an exam	3.1.1	Exam creation (back)	5
				3.1.2	Exam creation (front)	3
				3.1.3	Api security	1
				3.1.4	Unit / behavioral Testing	2
				3.1.5	Api documentation	1
		3.2	As professor , I can create questions	3.2.1	Open question creation (backend)	3
				3.2.2	Open question (frontend)	2
				3.2.3	Api Security	1
				3.2.4	Unit / behavioral testing	2
				3.2.5	Api documentation	1
		3.3	As a student I can pass quiz's	3.3.1	Pass quiz (back)	5
				3.3.2	Pass quiz (front)	3
				3.3.3	Unit / behavioral testing	2
				3.3.4	Api security	1
				3.3.5	Api documentation	1
		3.4	As a student I can pass a final exam	3.4.1	Pass final exam (backend)	3
				3.4.2	Pass final exam (front)	2
				3.4.3	Unit / behavioral testing	2
				3.4.4	Api security	1
				3.4.5	Api documentation	1
		3.5	As teacher, I can preview final exam submissions	3.5.1	Preview list of exams (backend)	5
				3.5.2	Preview list of exams (frontend)	3
				3.5.3	Preview a specific exam (frontend)	3
				3.5.4	Unit / behavioral testing	1
				3.5.5	Api security	1
				3.5.6	Api documentation	1
		3.6	As teacher, I can set marks to open questions	3.6.1	Set mark for a open question (backend)	2
				3.6.2	Set mark for open question (front	3
				3.6.3	Unit / behavioral testing	2
				3.6.4	Api security	1
				3.6.5	Api documentation	1
		3.7	As student I can preview my exam or quiz submission and mark	3.7.1	Get a specific exam details (backend)	3
				3.7.2	Display test result	3
				3.7.3	Unit / behavioral testing	2
				3.7.4	Api security	1
				3.7.5	Api documentation	1

Table 4-1 Release 2 backlog

III- Modeling

1- Exam Use case

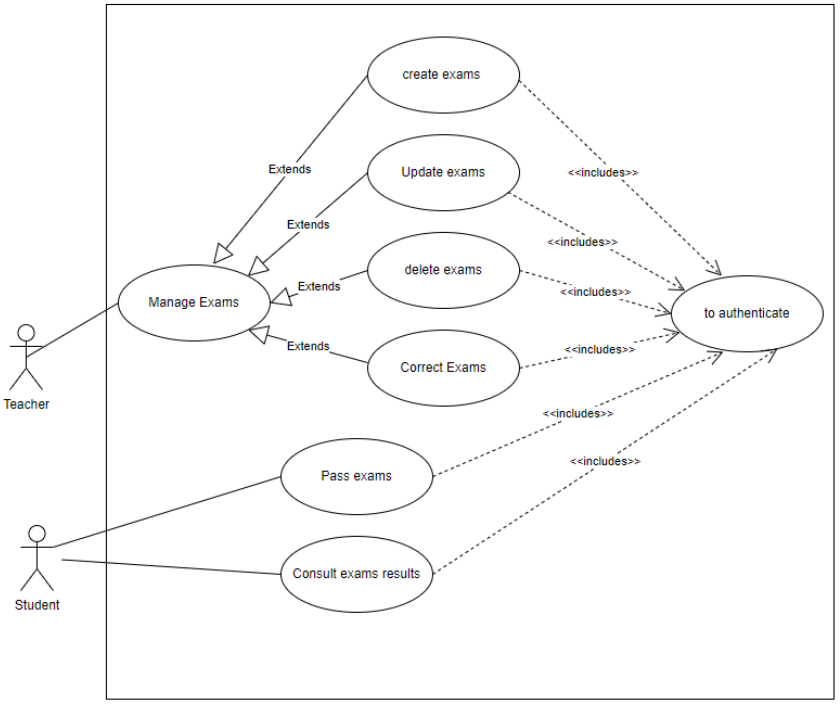


Figure 4-1 exam use case

2- Inscription form request use case

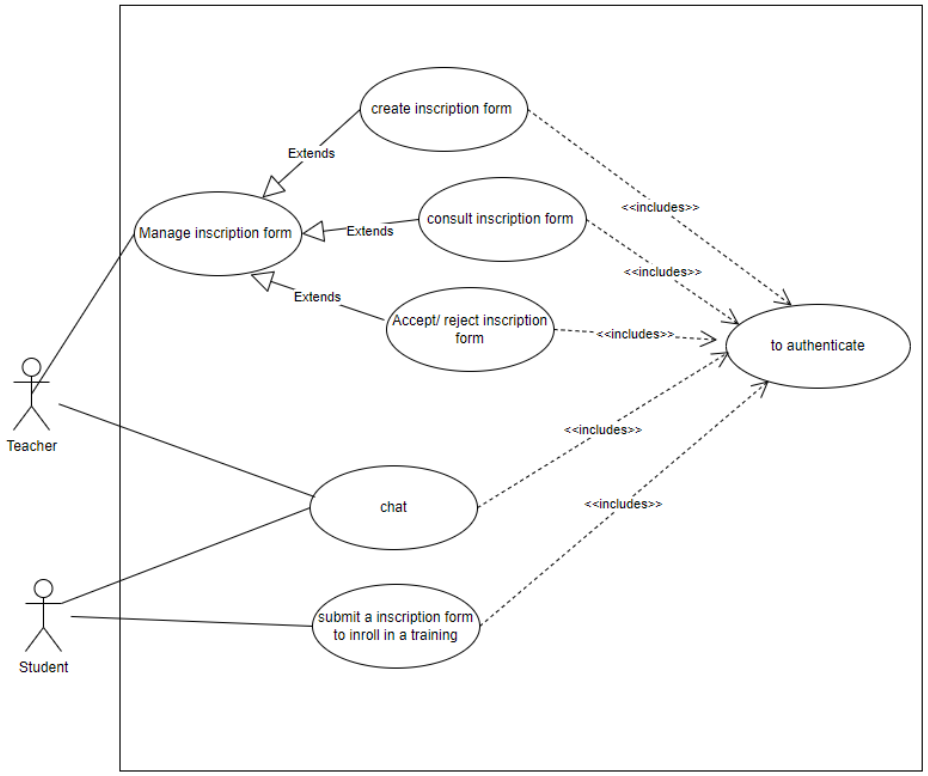


Figure 4-2 inscription form use case

3- Exam Class Diagram

Figure 4-3 exam class diagram

IV- Release deliverable

1- Creating the inscription form

Some courses need skills to enroll, that's why the teacher can create a form to evaluate students and choose whom have the suitable profiles

Use Case	Creating inscription form
Actors	Professor
Hypothesis	A professor can create an inscription form to answer in order to enroll in a training
Pre-conditions	During creating a training, a professor can create a form to evaluate students before enrolling into a training
Initialization	This use case starts when the teacher chooses to set course with an inscription form
Description of the scenario	- The teacher starts creating a training - The teacher clicks on inscription button - The teacher choose answering a inscription form in training inscription condition
Post-conditions	After completing this action , answering the created form will be a condition to take this course

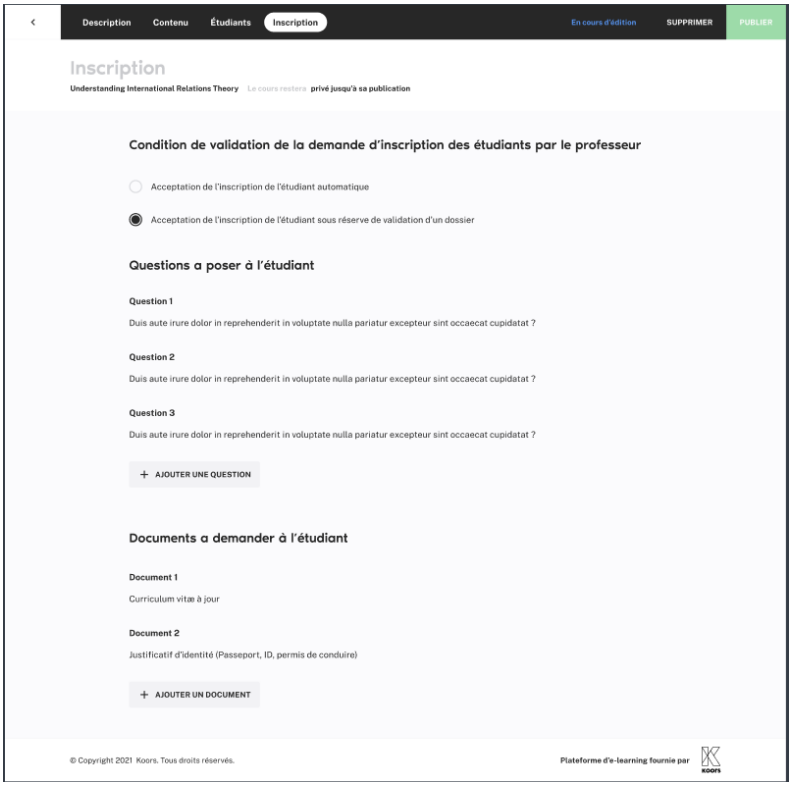


Figure 4-4 Inscription form creation

To review submissions, the teacher needs to access the list of submissions, consult the profile of student, his answers to the form and finally choosing if he will be accepted or refused

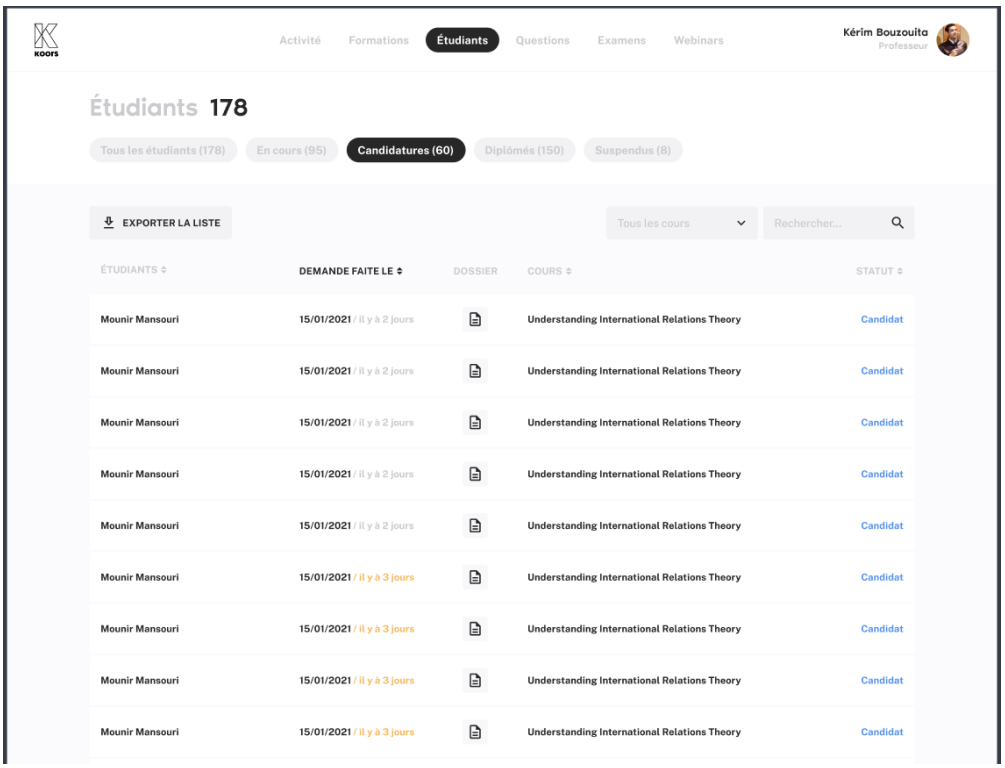


Figure 4-5 Student list page

2- Creating a exam

In order to evaluate the level of students, the professor can create quiz's and exams that they will be passed during tacking the course

Use Case	Creating an exam
Actors	Professor
Hypothesis	This professor wants to create a quiz/ final exam
Pre-conditions	any professor can create a quiz during the creation of a course
Initialization	This use case starts when the professor clicks on create exam button
Description of the scenario	- The professor clicks on create exam button - The professor will choose the needed type of evaluation (quiz or final exam) - The professor adds the questions and gives the exam a title and duration - The professor saves the exam
Post-conditions	The quiz/ final exam will be added to the database and to the content of this course , path or session

Table 4-2 create exam textual use case

a- Sequence diagram

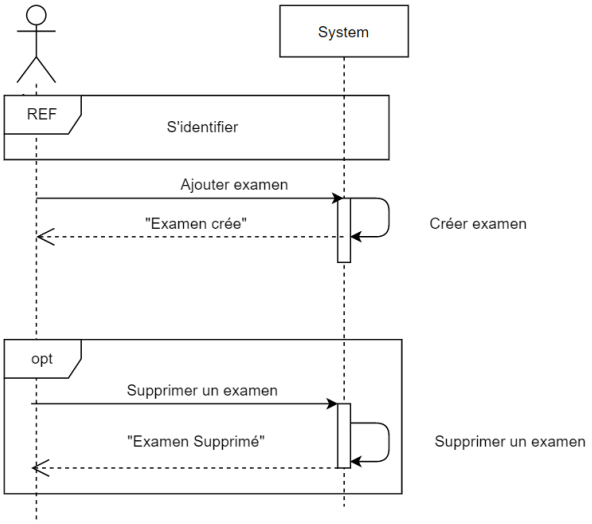


Figure 4-6 System sequence diagram

After clicking on create exam button, a modal will pop out in order to let the professor choose the type of the exam to create and to choose between a simple quiz or a final exam

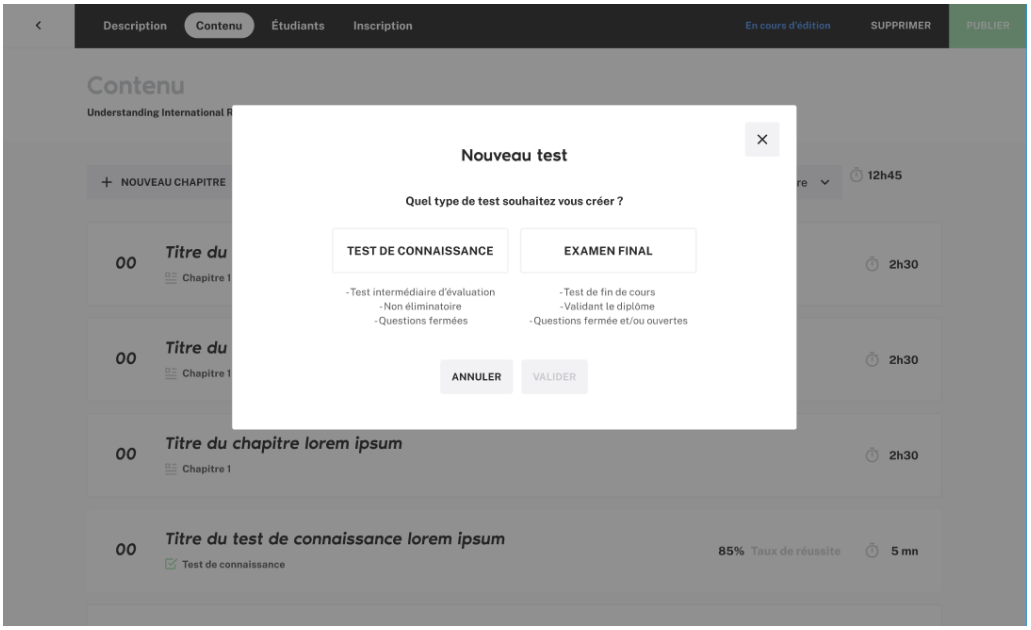


Figure 4-3 create exam step 1

After choosing the type exam, the professor will be redirect to exam's dynamic form where he will be able to create his exam using a dynamic form by setting the exam name, duration and creating exam questions

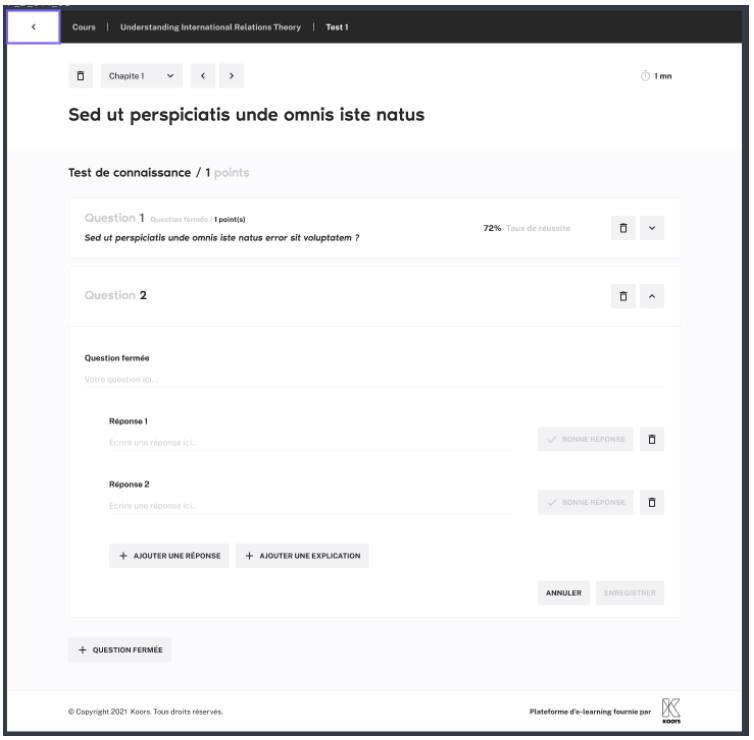


Figure 4-7 create exam step 2

3- Exam update

After creating the exam, the professor can consult the exam to review it before publishing the training in order to update the exam content, title or duration. For the scale, a quiz have only 1 point as a mark and for the open ones, he can choose a scale mark between 1 to 5 points

a- Sequence Diagram

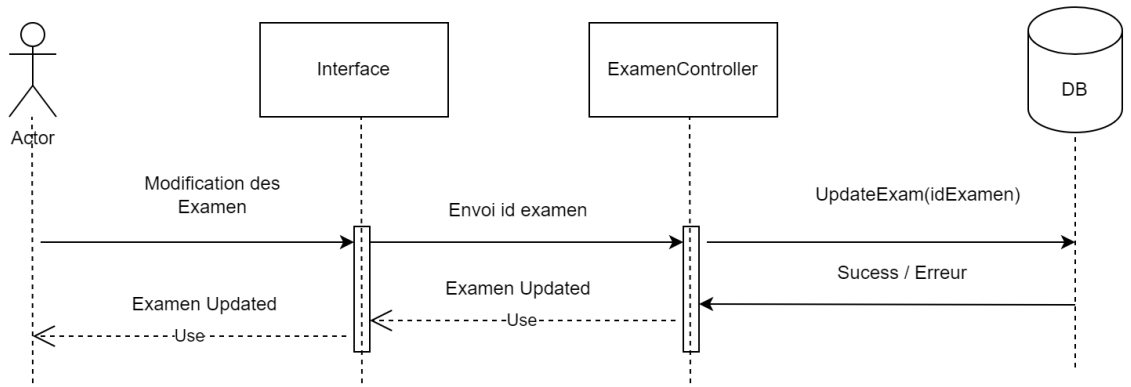


Figure 4-8 update exam sequence diagram

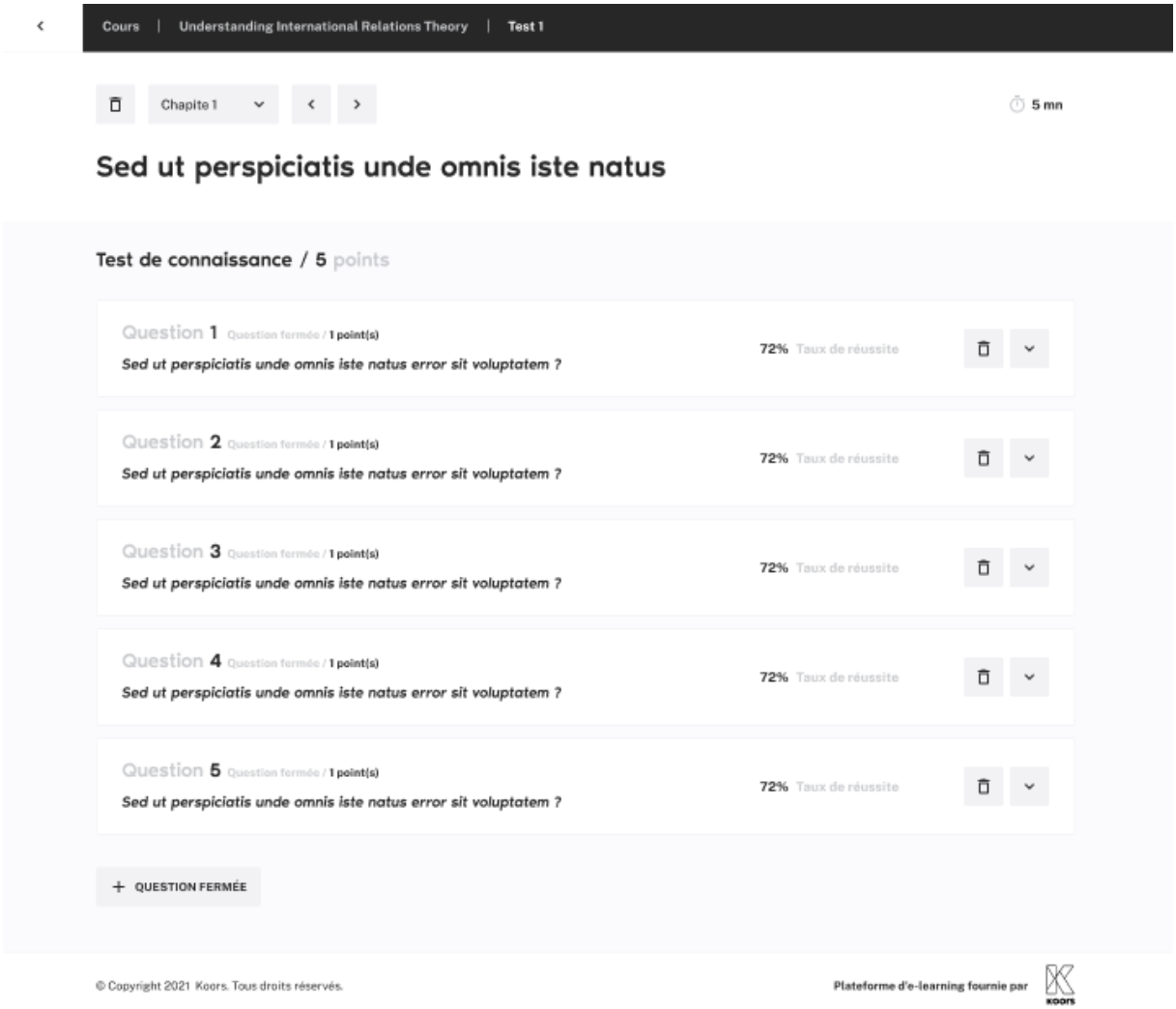


Figure 4-9 update exam screen

4- Passing an exam

During pursuing a training, the student must respond to quiz's or exams in order to have a diploma after completing the content of this course. After finishing the test, a screen will appear to show the results. In a case of final exam, a review screen will be appeared until the final exam will be evaluated by the professor

Use Case	Passing exam
Actors	Student
Hypothesis	A student wants to pas his quiz or final exam
Pre-conditions	During taking a course, student must answer to quiz/ final exam
Initialization	This use case starts when the student starts the exam
Description of the scenario	- The student clicks on start exam buttons - The student answers on questions
Post-conditions	After completing the tests, students' responses will be stored in the database and the student will be redirected to the progress course page

Table 4-4 passing exam textual use case

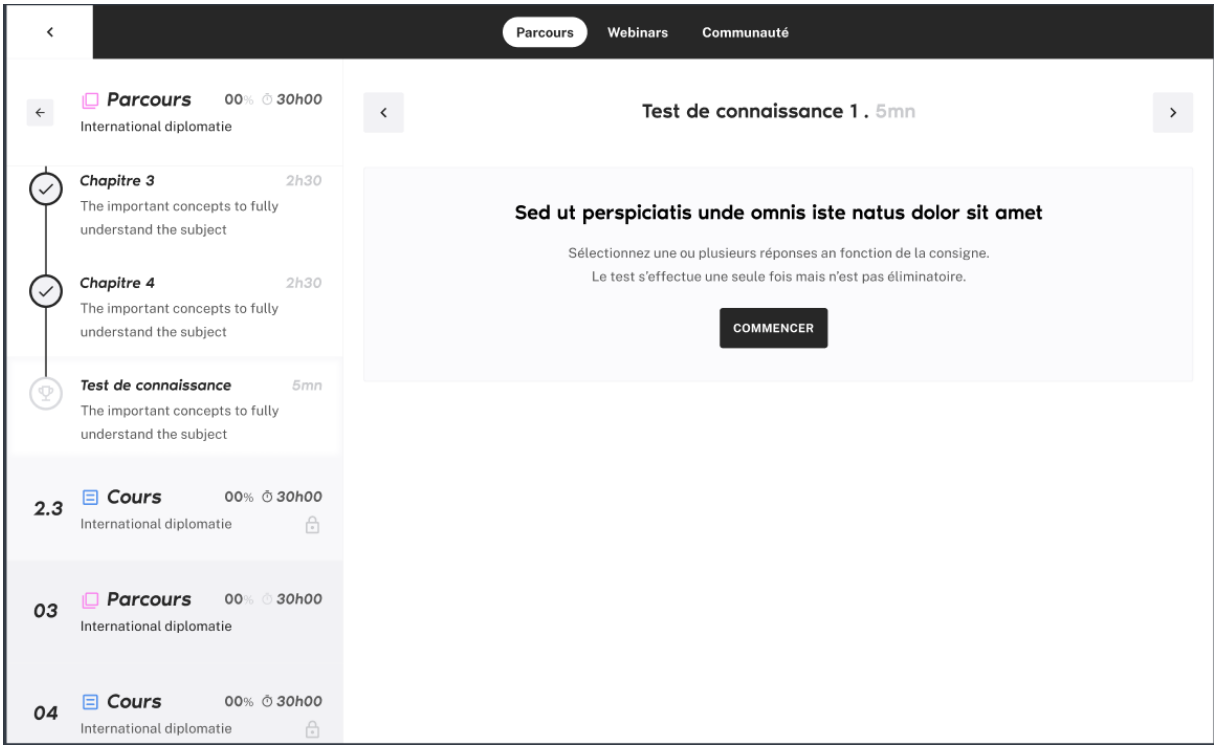


Figure 4-10 pass exam screen

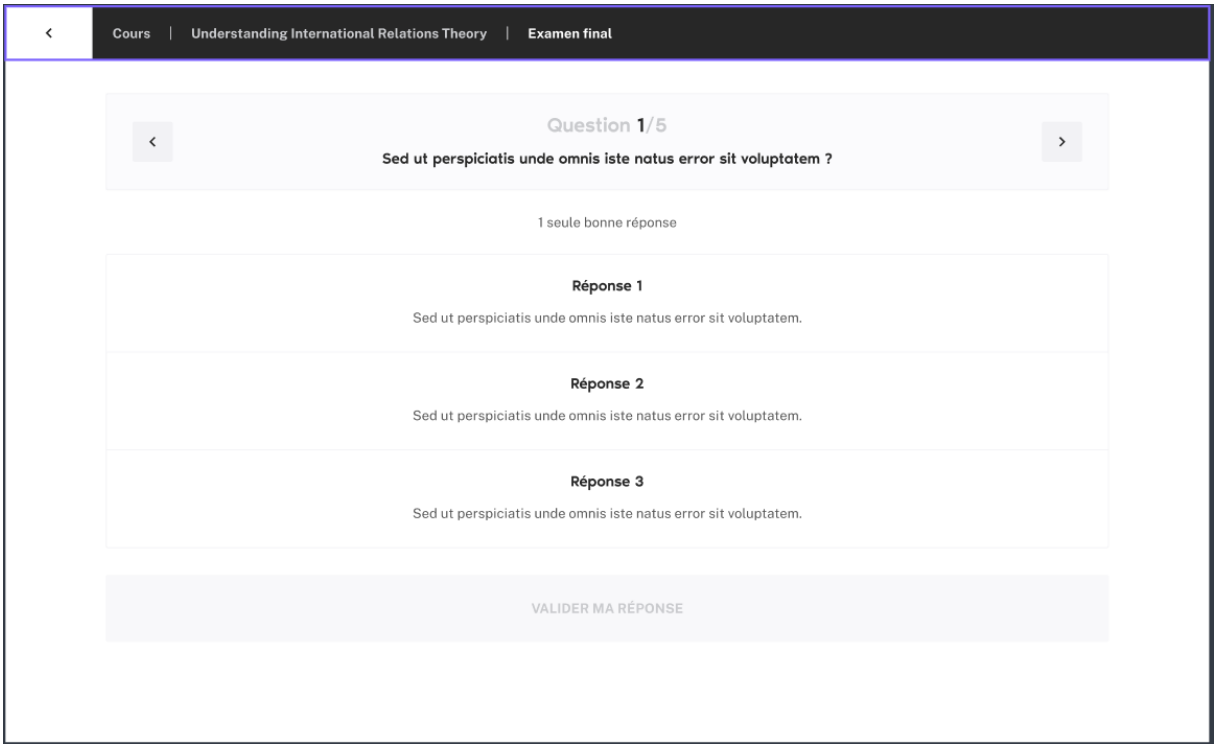


Figure 4-11 pass quiz

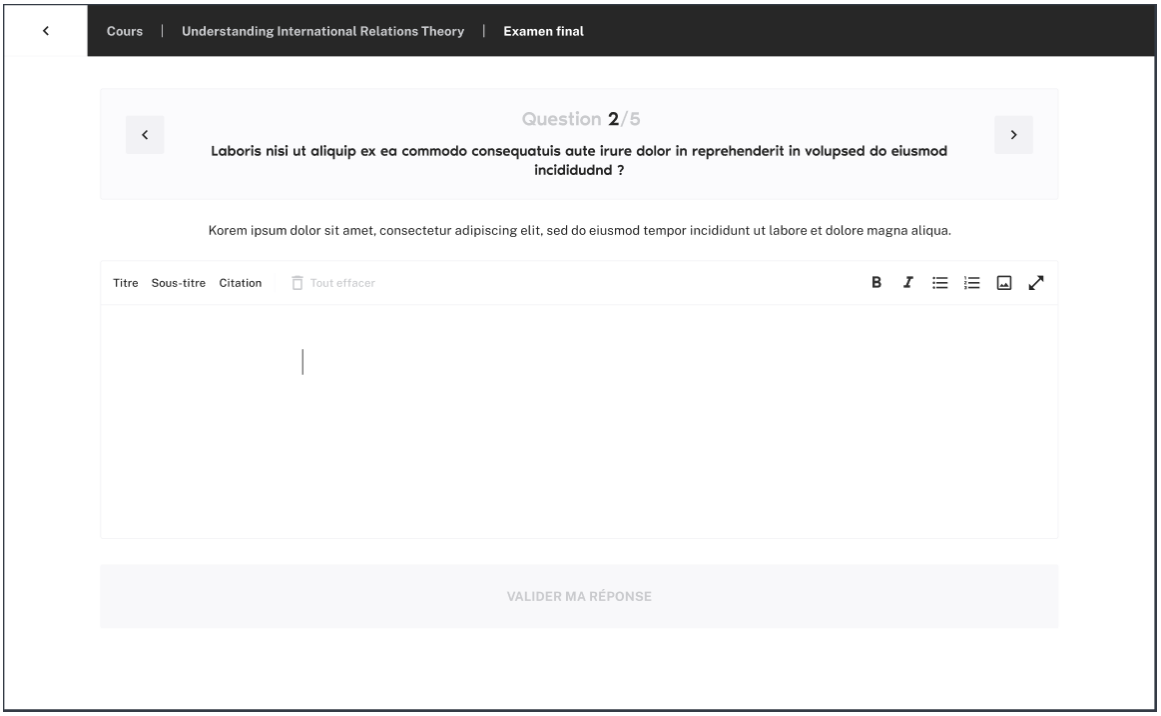


Figure 4-12 Respond to open question

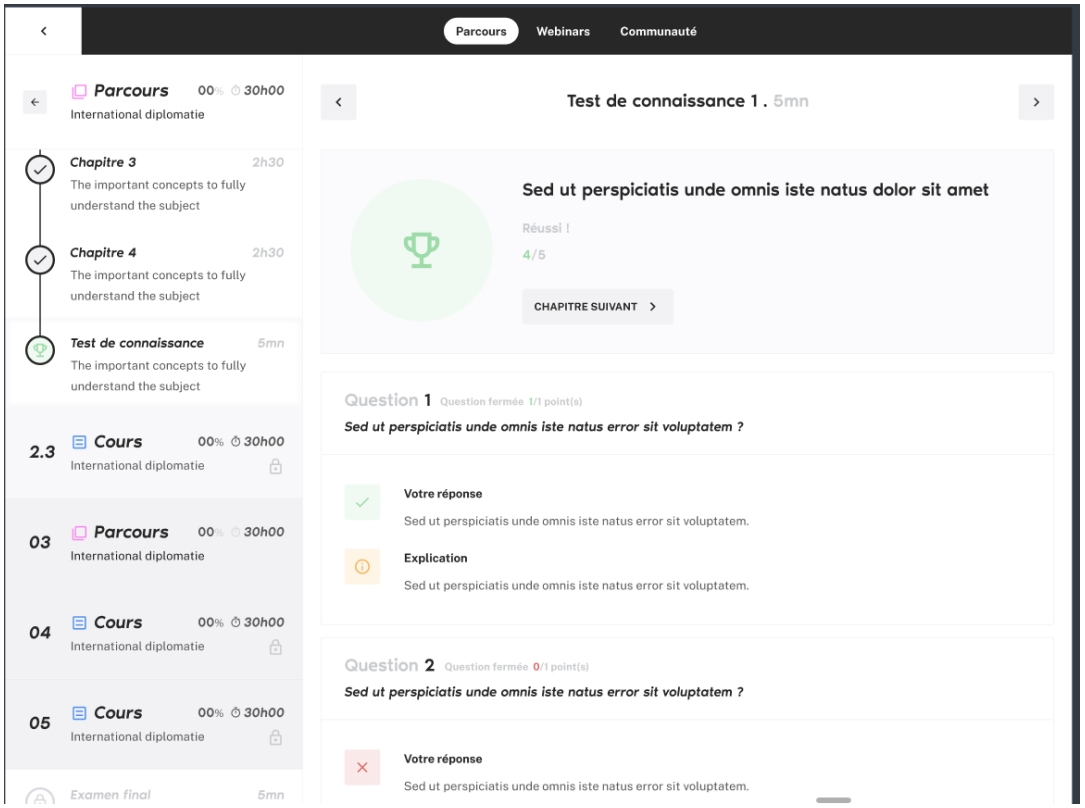


Figure 4-13 display exam result

The student can pass successfully every evaluation form unless his mark is grater then the half of the scale note of the exam or else it will be considered failed

5- Final exam correction

In case of final exam, some questions must be corrected by the professor. He can have access to this section by clicking on Exams section in the navbar. After that, the list of exams will appear with the possibility of choosing the status of the exam (exams to review / completed exams or all exams)

The table below show exams status with it submission date, the name of the student and submission date and it be filtered by the training name to facilitate the evaluation process by the teacher and finally it can be exported as a csv file

a- Textual use case

Use Case	Consulting final exams list
Actors	Professor
Hypothesis	A professor wants the list of final exams
Pre-conditions	The professor wants the list of final exams
Initialization	This use case starts when the professor wants to view the list of final exams
Description of the scenario	- The teacher clicks on the exam button in the navbar
Post-conditions	After completing the scenario, the list of final exams will be appeared

Table 4-5 consult final list textual use case

b- Sequence Diagram

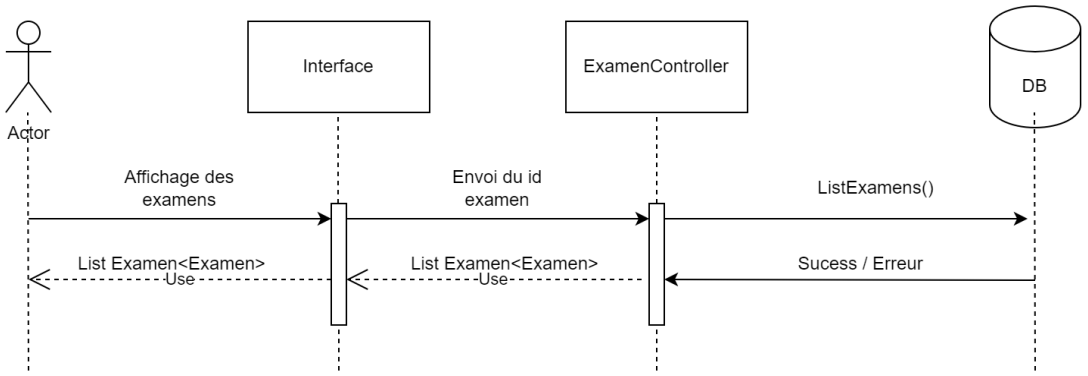


Figure 4-14 display exam sequence diagram

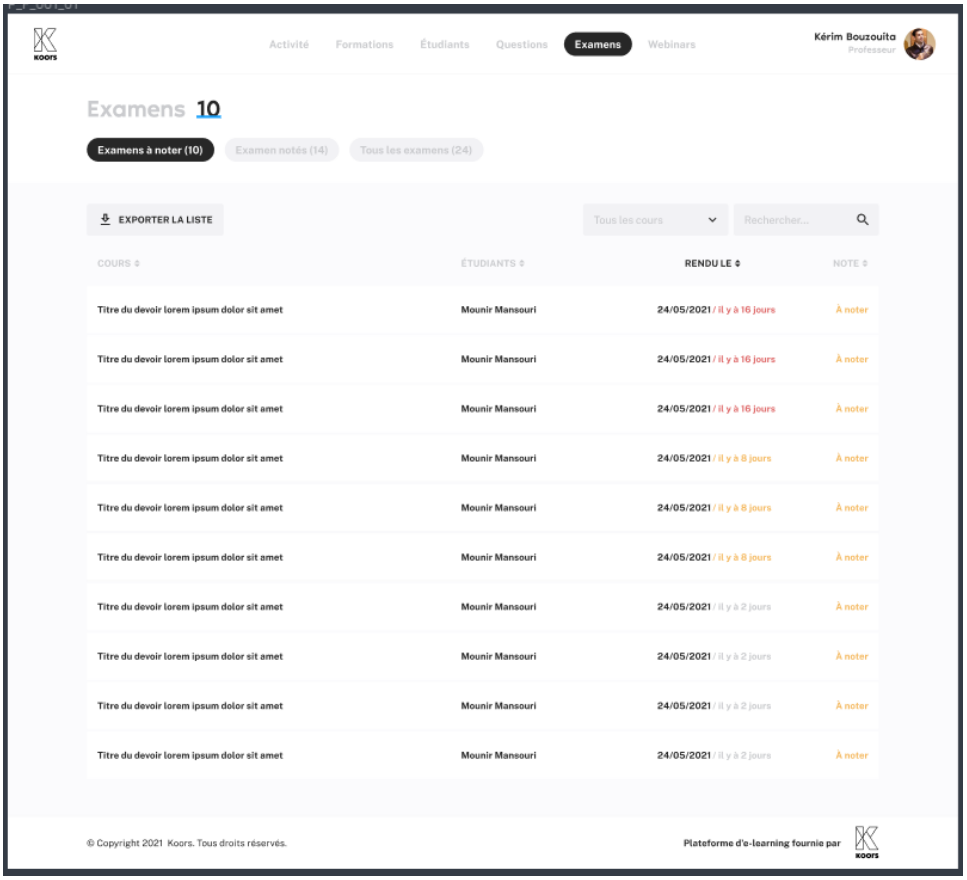
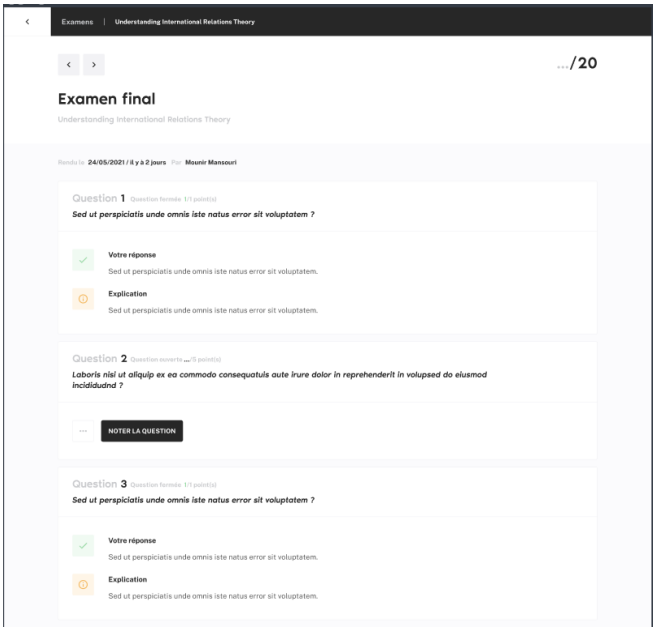


Figure 4-15 display exam list

To correct an exam, the teacher clicks on the exams, choose a question to correct and giving mark to students



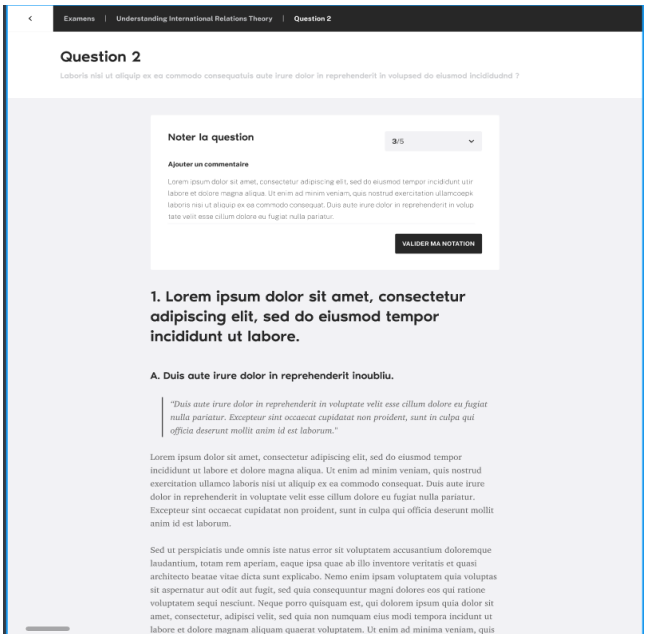


Figure 4-16 correct exam screens

If all questions of the exam are evaluated, his final mark will appear, and its status will change to completed with the student mark

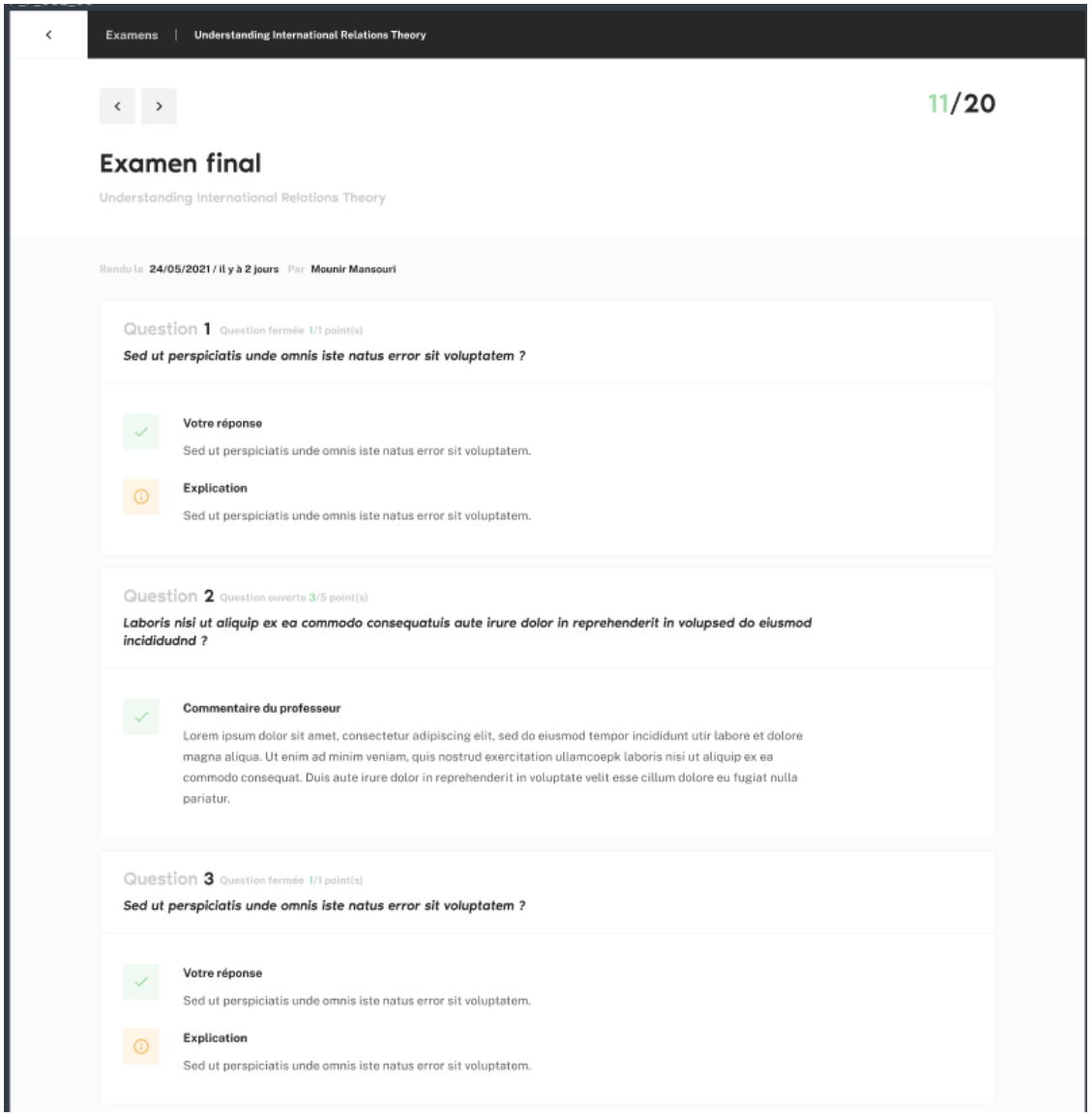


Figure 4-17 final exam after correction screen

V- Conclusion

In this release, we have developed the frontend and the backend part of the continuous evaluation module and inscription form, also we have corrected declared bugs

Chapter 5 : Release 3: Payment Module

I- Introduction

In this release, we're going to develop a payment module to give the student the chance to enroll in paid trainings of the platform and our choice was Paymee payment engine to make payments in Tunisia

II- Release Backlog

	Feature	ID	User Story	ID Task	Task	Estimation
4	Payment Module	4.1	As director, I can fix course, session path price	4.1.1	Fix content price (back)	5
				4.1.2	Fix content price (front)	3
				4.1.3	Unit / behavioral testing	2
				4.1.4	API security	1
				4.1.5	API Documentation	1
		4.2	As director, I can manage courses price	4.2.1	Manage content price (back)	5
				4.2.2	Manage content price (front)	3
				4.2.3	Unit / behavioral testing	2
				4.2.4	Securing the API	1
				4.2.5	API documentation	1
		4.3	As director, I can set payment method for his school	4.3.1	Choose payment method (back)	3
				4.3.2	Choose payment method (front)	3
				4.3.3	Unit / behavioral testing	2
				4.3.4	API Documentation	1
				4.3.5	Securing API	1
		4.4	As director, I can consult payment history	4.4.1	Consult payment history (back)	5
				4.4.2	Consult payment history (front)	3
				4.4.3	Unit / Behavioral testing	2
				4.4.4	Securing API	1
				4.4.5	API documentation	1
		4.5	As a student, I can pay a training with paymee	4.5.1	Paymee payment(Back)	5
				4.5.2	Paymee payment(Front)	3
				4.5.3	Unit / Behavioral testing	1
				4.5.4	Securing API	1
				4.5.5	API documentation	1

Table 5-1 backlog release 3

III- Modeling

1- Payment use case diagram

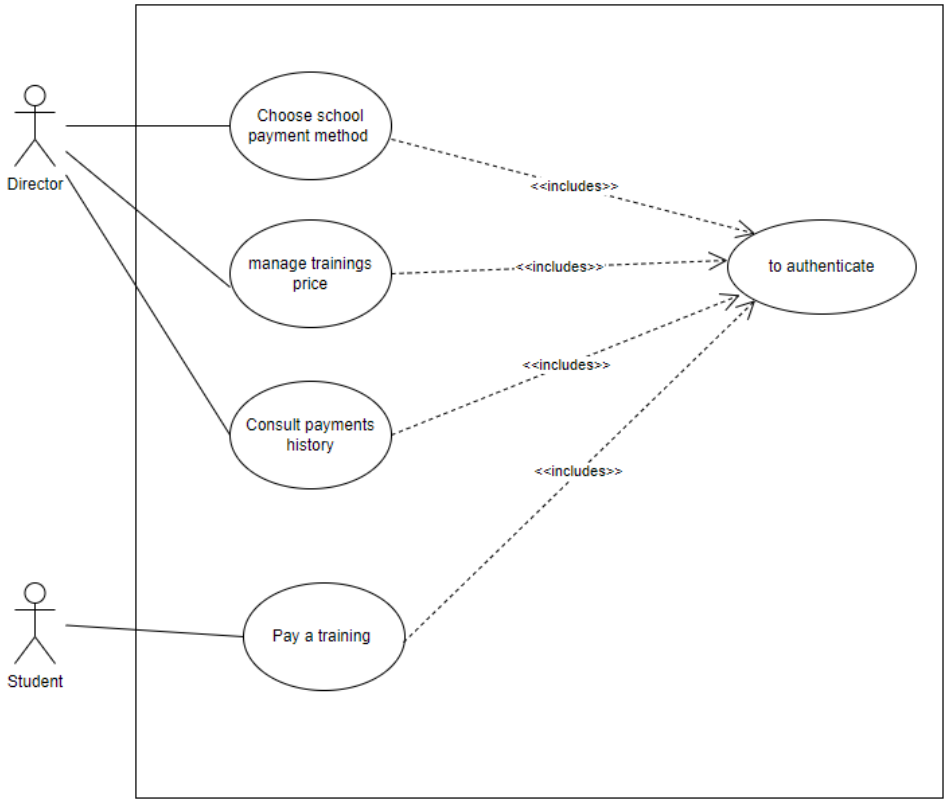


Figure 5-1 Payment use case

2- Payment class diagram

Figure 5-2 Payment class diagram

IV- Release deliverable

1- Setting payment method

In order to accept payments, the director must choose a payment method and link his account to the platform, In KOORS case, to accept payments in Tunisia , the director must choose Paymee as payment engine and to have payment from other countries , Strip payment solution is right option to choose

Use Case	Setting payment method
Actors	Director

Hypothesis	A director wants to set a payment method
Pre-conditions	During configuring the school, the director must choose a payment engine to be able to have paid transactions
Initialization	This use case starts when the director clicks in Payment section or while setting his first paid training
Description of the scenario	- The director clicks in school button in the navbar - The director clicks in payment section in school info - The director chooses a payment engine - The director saves his choice by clicking on save button
Post-conditions	After choosing a payment engine, director choice will be stored in the database and the currency of paid training will be displayed

Table 5-2 Setting payment method textual use case

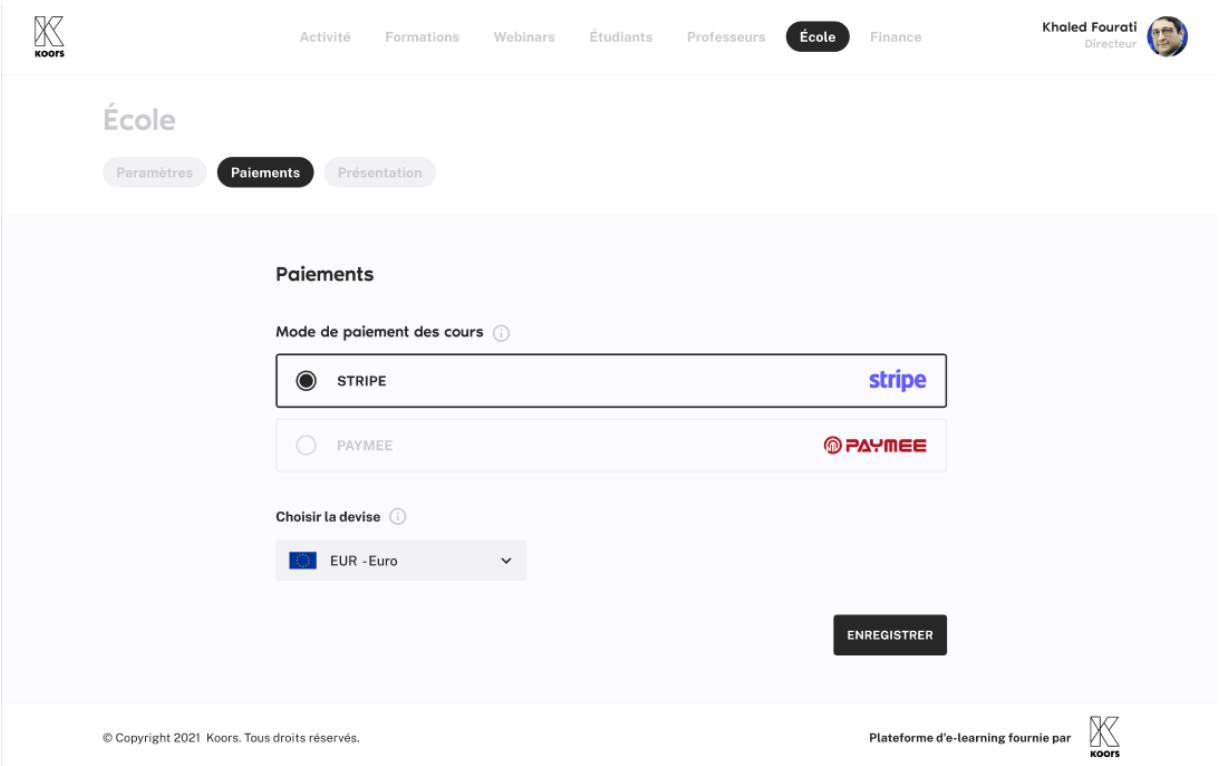


Figure 5-3 setting payment method screen

2- Setting price for training

As director, He can set if the course to publish is free or with fees. First, he chooses the payment method, then he set the price to every paid course, path or session

He can also manage the price after consulting the list of paid courses in finance section with the history of price for every course with the possibility to make the course free instead of paid

Use Case	Setting training price
Actors	Director
Hypothesis	A director wants to set training price

Pre-conditions	During reviewing a training, the director sets the training price
Initialization	This use case starts when the director clicks on paid button
Description of the scenario	- The director clicks on paid option for inscription type - The director sets the price - The director saves his choice and approves the training
Post-conditions	After approving the training, the data will be stored in the database and the director will be redirected to the list of courses

Table 5-3 setting training price textual use case

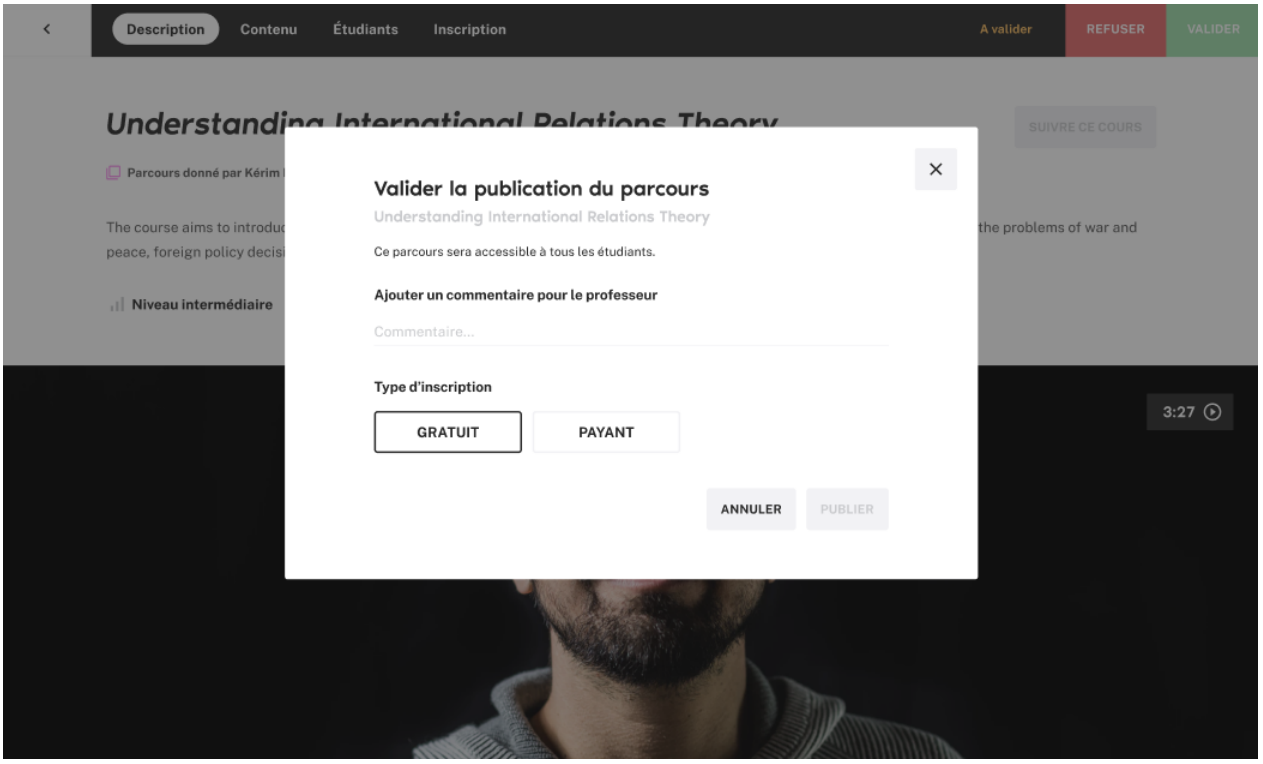


Figure 5-4 setting training price

3- Payment history

In order to consult student payments history, the director must go to finance section displayed in the navbar, A table containing the details of payments transactions will be displayed with two views: the first using the name of training and the second one using the history of sales by student's transactions

Use Case	Consulting payment history
Actors	Director
Hypothesis	A director can consult list of payments
Pre-conditions	The director wants to view his school financial history

Initialization	This use case starts when the director clicks on finance button in the navbar
Description of the scenario	- The director clicks on finance button in the navbar
Post-conditions	After displaying the list, the director can view the course financial history

Table 5-4 payment history screen

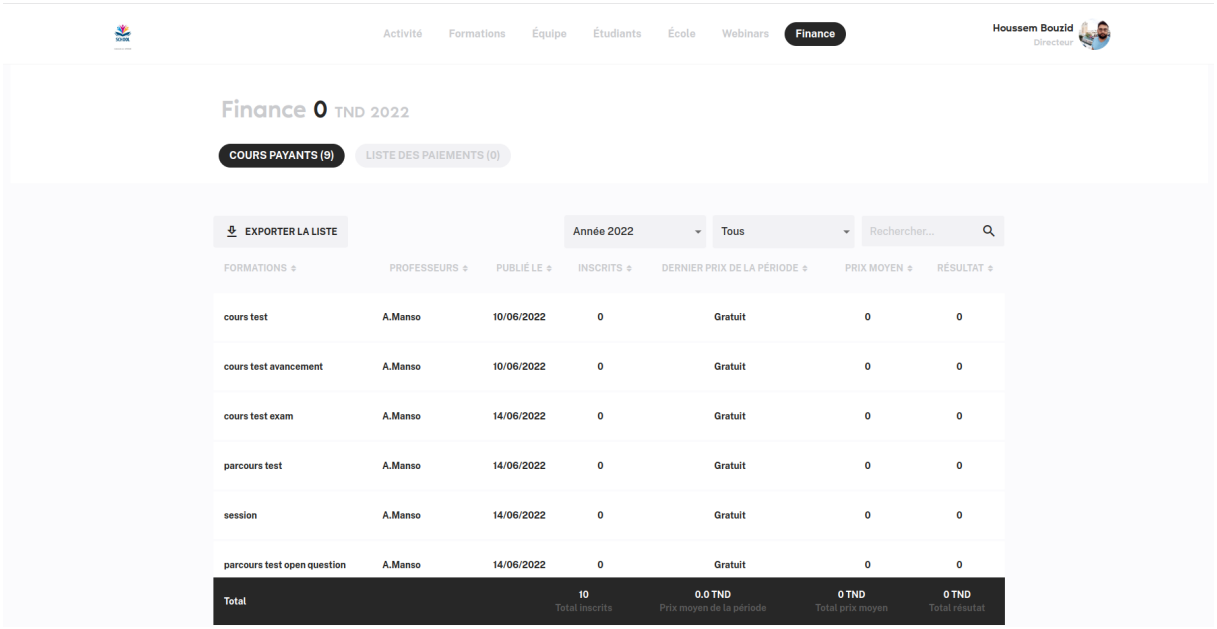


Figure 5-5 payment history screen

Payments can be filtered by multiple filters such as month, name of professor or the name of training. The footer of the table displays an overview for the table rows that shows number of enrolled students and total of student's transactions

4- Training price management

As a director, he can manage training price, view it history and statistics and change the price or change it to free training

Use Case	Training price management
Actors	Director
Hypothesis	Director can manage the training price
Pre-conditions	During clicks on a specific course in the financial part table
Initialization	This use case starts when the director clicks on a course
Description of the scenario	- The director clicks on finance button in the navbar - The director clicks on a specific course

	- The director sets the new training marks - The director saves the new price by clicking on save button
Post-conditions	After clicking on save button, the new price will be stored in the database and the new price will be displayed for students inscribed in the school

Table 5-5 training price management

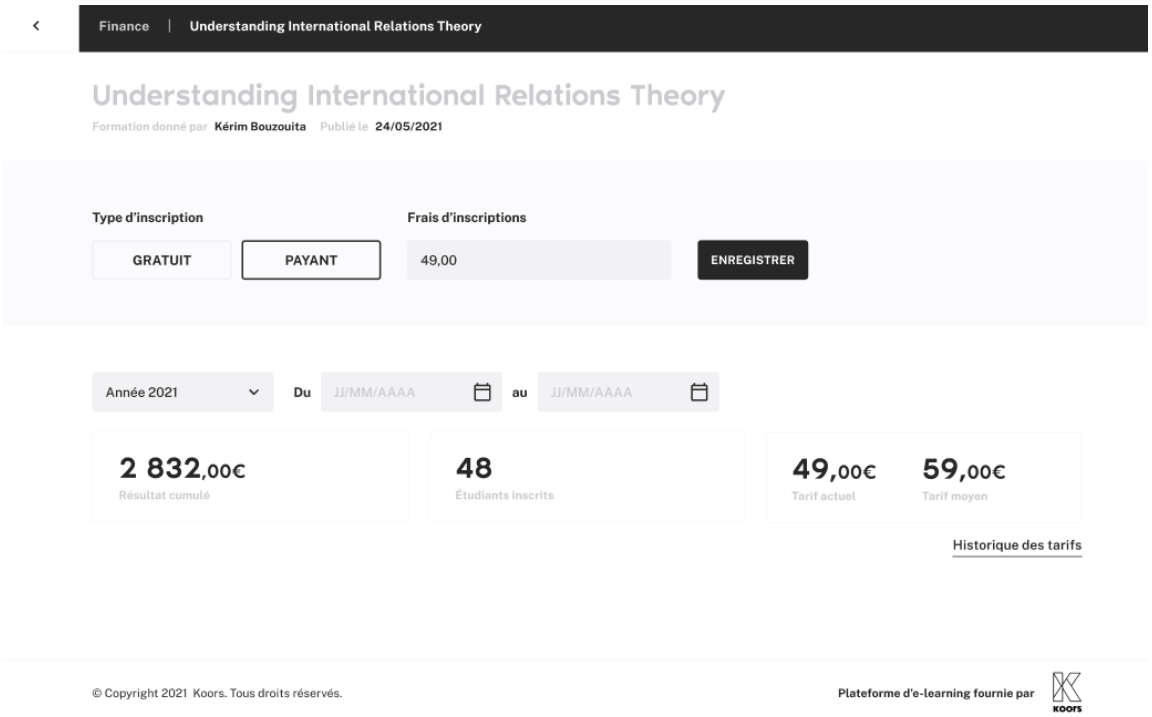


Figure 5-6 training price management screen

5- Make payments with paymee

As a student, I can buy paid training with Paymee in Tunisia using my credit card or Paymee account

Use Case	Paid training payment with paymee
Actors	Student
Hypothesis	Student can pay training fees with Paymee
Pre-conditions	During clicks on a specific paid course
Initialization	This use case starts when the director clicks on a course
Description of the scenario	- The student clicks on catalog section - The student clicks on a specific paid training - The student clicks on enroll course

	- The student pays training fees with Paymee
Post-conditions	After Payment, the student will be enrolled in the selected course and he will be redirected to the start training page

Table 5-6 pay training with paymee textual use case

V- Conclusion

In this release, we have developed the frontend and the backend part of the Paymee payment module, also we have corrected declared bugs

Conclusion and perspective

In this final project, carried out at Slashup Studio, we've designed and developed a web application in e-learning. As a methodology, the choice fell on Scrum as a working method using React for the Front-end with redux, SpringBoot and NodeJs for the Back-end and PostgresSql for the database.

As a start point, we've began with study of existence to specify our needs and modules. Afterwards, we've started the modeling and the design part in order to have a clear vision on the platform.

Next, we started the development phase after dividing the work as releases and by the end of each release, we push the new functionalities into production after fixing bugs.

Integration into a professional work team was very fluid and working with them was a pleasure even when we have many tasks to accomplish.

And now after developing our platform, it would be worth improving our functionaries to improve user experience, integrating Kafka and micro frontend in our upcoming versions

This presented work, is developed and done in order to obtain Software engineering degree at ESPRIT school of engineering as a graduation internship

List of references

[1] https://medium.com/chingu/a-short-introduction-to-the-scrum-methodology-7a23431b9f17

[2] https://medium.com/@joespinelli_6190/mvc-model-view-controller-ef878e2fd6f5

[3] https://medium.com/swlh/microservices-architecture-from-a-to-z-7287da1c5d28