UCL Université catholique de Louvain

École polytechnique de Louvain (EPL)

epl

# Adding a module to the Oscar educational website

## allowing teachers to create new types of interactive exercises for their students

Dissertation presented by
**Antoine HABRAN , Nicolas TONON**

for obtaining the Master's degree in
**Computer Science (Master[60])**

Supervisor(s)
**Laurent FOURNY, Kim MENS**

Reader(s)
**Laurent FOURNY, Kim MENS , Chantal PONCIN**

Academic year 2017-2018

# Abstract

The non-profit organization, *Eureduka*, is developing *Oscar* [1], an educational website intended to support the work of teachers and to improve the learning experience of students (figure 1). In its actual state, the website provides tools to evaluate the knowledge and skills of students on precise questions. It also provides educational resources adapted to the student's level. One of the key concepts of Oscar is to organize the skills in a tree structure where each node represents a specific skill. The tree abstraction represents the order in which the student should master the skills where a "child node" should be acquired after its parent. For example, the skill "multiplying integers" should be mastered before trying to multiply decimals.

In this context, we were asked by the Oscar team to provide a new module whose purpose is to assess mathematical reasoning skills for its website.

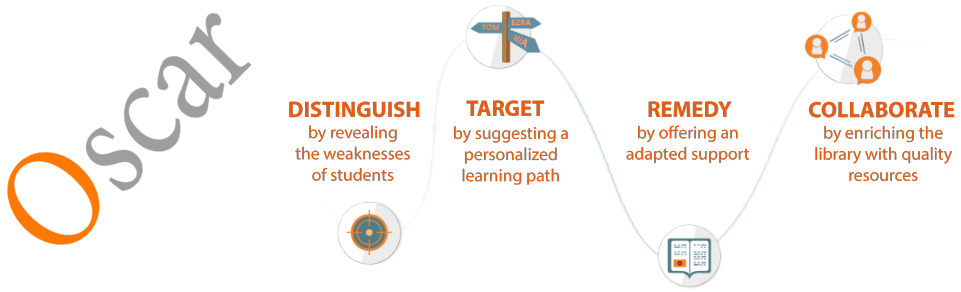



Figure 1: Oscar's learning process [1]

# Contents

# Introduction

In the context of our master thesis, we were asked to create a new module to be integrated into Oscar in which students could be evaluated on reasoning exercises composed of multiple steps. On one hand, this module should allow teachers to create new reasoning exercises in which they can define the different steps composing a correct sequence. On the other hand, it should allow students to try to solve the exercise by organizing the set of elements created by the teacher and presented to him in a random arrangement.

## Approach

The basic idea of the client was to allow teachers to create elements through a graphical interface and to fill them with various contents. Then he would give the elements a certain order and finally, after validation and submission, its students could access the exercise. He would see the elements randomly displayed and would have to re-arrange them in the order the teacher intended. However, this was only the basis of the original idea. The further the module advanced, the more we expanded this idea with the client.

## Objectives

The goal of the module is twofold. First, it should allow students to train on the theory covered by the exercise. Second it should allow the teacher to evaluate the student's skills and to get a better understanding of the student's level.

## Limitations

Since the Oscar website is under continuous development, some of its parts are not working perfectly. As a consequence, the integration part will be more difficult than if it was not since we have to deal with possible missing functionalities of the website. Moreover, some parts of our module might be implemented but would not be functional until the needed components of the website are not either.

# Chapter 1

# Theorizing the requirements of a reasoning assessment module

In order to build a reflexion to identify possible tracks of implementation, we will take interest in several topics surrounding the desired module and its goals.

1. First, we need to understand the basics of reasoning and particularly mathematical reasoning.
2. After that, we will see how reasoning is learned.
3. We will then be able to find out how to evaluate this mathematical reasoning.
4. We will therefore have the necessary means to identify some needed specifications of the desired module.
5. Eventually, a proposition of a theoretical implementation will be given.

All the topics discussed in this chapter are the work of a personal questioning about the stakes surrounding mathematical reasoning with the help of existing research. It will aim to give a non-exhaustive critical view to induce a logical path of development for the desired module.

## 1.1 What is mathematical reasoning ?

Reasoning is the work of taking or receiving information and using knowledge and past experiences to compare this information to draw conclusions. This skill can be taught and improved upon.

Thus, mathematical reasoning is using mathematical knowledge on new informations to draw mathematical conclusions. For example, a mathematical reasoning exercise could be : "Let A be a set of numbers, classify those numbers in sets with different properties".

"Mathematical reasoning is important. It's the critical skill that enables a student to make use of all other mathematical skills. With the development of mathematical reasoning, students recognize that mathematics makes sense and can be understood. They learn how to evaluate situations, select problem-solving strategies, draw logical conclusions, develop and describe solutions, and recognize how those solutions can be applied. Mathematical reasoners are able to reflect on solutions to problems and determine whether or not they make sense." [2]

## 1.2 How to learn reasoning ?

There are several ways to learn and improve one's reasoning [6] :

1. Expression and freedom : By giving students the opportunity to express themselves and to argue their choices, we allow them to take a personal part in the daily life's different activities and to explain their believes and reflexions to other ones.

2. Confrontation and adaptation : Another way to improve one's reasoning could be to encourage confrontations between multiple reflexions in different situations, topics to induce a process of validating the reasonings. To argue about a topic is to extract different informations building the subjects and apply a certain reasoning to keep only the relevant information and discard the rest.

3. Research and autonomy : As the reasoning is a research, we can build an environment in which students need on their own to investigate topics in which they give interests to build reflexion paths.

4. Analyzing and questioning : By questioning and analyzing the world around ourself, we enable critical reasoning. This create a philosophy of learning and improving the general understanding composing our daily life.

## 1.3 How to evaluate mathematical reasoning ?

By definition, in a reasoning exercise, the interrogator gives a set of information and a reasoning question that the student will analyze with his knowledge to give an answer.

The goal of a reasoning exercise is to assess the student's reasoning skill and skill's knowledge. However there is an infinite quantity of subjects for exercises. Reasoning exercises can apply to every skill and its set of knowledge. Depending on the phrasing of the question, the type of expected answer and its format, the resulting exercises can be very different.

In order to give an answer, the student will use his reasoning capabilities. Although, there are different types of reasoning, everyone has their own. If we ask a student to classify numbers, he can start from the beginning of the list, from the end, from the middle, ... there are many possibilities.

In this type of exercise, there are two endogenous variables, directly linked to the content of the exercise, the information given and the question with the expected answer, and there are two exogenous variables, linked to the environment of the exercise, the knowledge of the student and its reasoning process, and his answer.

We can remark that the endogenous and exogenous variables have infinite possibilities. Even if the endogenous variables can be limited to a particular set of questions/answers and a set of knowledge, the exogenous variables will always be infinite.

## 1.4 Requirements of the desired module

Thanks to the analysis of reasoning and mathematical reasoning, we can extract requirements for the desired module.

The desired module can limit the student's ability of reasoning with its own environment. As the reasoning is not just a computation of complex or basic operations but the computations of logical links between various skills, the liberty of actions is a key factor to consider but can also limit the teacher's ability to assess the students correctly.

Indeed, to have the perfect environment, we need to allow the teachers to create, on one hand, an infinite number of different questions, on the other hand, for an infinite number of possible subjects. But, we also need to allow the student to answer the exercises in any possible way, that is to say an infinite number of answers, in an infinite number of formats. However, this is obviously very difficulty or even impossible to achieve.

Nevertheless, we can create a non-exhaustive list of properties that a reasoning assessment tool should have :

- to provide flexibility to the teachers. The teachers need liberty to write the most appropriate question that will ask the students to use correctly his knowledge for the skill.
- to provide flexibility to the students. As there is no single way to reason, a student shouldn't be impacted by the way the answer needs to be formated.
- to be easy to use for the teachers and students.

## 1.5 Proposition of a theoretical implementation of the desired module

In a school, at university, the use of paper provides the teacher a way to ask any question that can be answered in a 2D format and at the same time, provide the student to give any answer of that format.

To recreate this flexibility, we can imagine the use of a white board tool where the student could, at his will but with some constraints impacted by the environment, grab or create objects predefined by the teacher and move them, write down his thinking, draw graphs and other figures, modify objects, ... It also gives the teacher a large variety of ways for the student to be tested.

However, while building such a tool would provide a very good environment for the students and the teachers, it would require a lot of development and a high-level of compatibility between the different options/sub-tools.

This idea can be partitioned in sub-problems by the use of templates. A template for a white board tool is a predefined state and use of the white board. It would restrain the liberty to some degree while still allowing the teacher to create creative question for a precise themed type of questions. And as the need for new ways of assessing methods is needed, new templates could be programmed into the module to fill those needs.

For example, we can imagine a question in which the student need to order a set of steps. This question could be translated in a pre-defined template like figure 1.1. Where the items are randomly placed on the left, waiting to be ordered in the cells on the right.

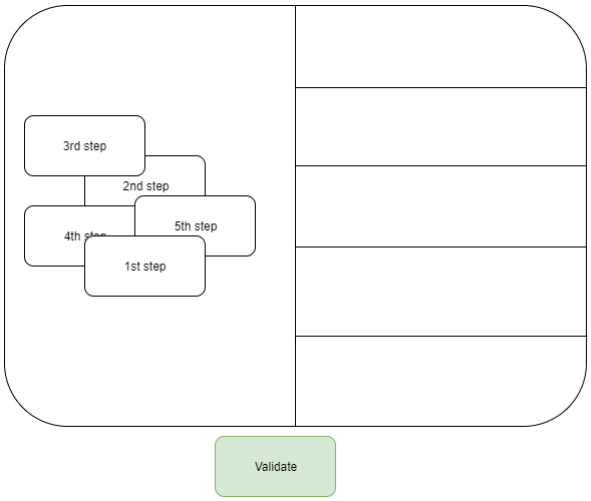



Figure 1.1: An example of template for ordering questions

# Chapter 2

# State of the Art

Nowadays, digital learning is becoming more and more predominant thanks to its easy access, diversity and other advantages. Various applications and websites already offers similar services to the desired module. In this section, we are gonna talk about such applications and websites and how they achieve digital learning.

## 2.1 Learning apps

*Learning apps* [8] is a website providing tools allowing teachers to create learning applications. In order to make it easier, the teachers can choose between multiple templates to create their own exercises. As we are asked to create a module to evaluate mathematical reasoning, three templates, Number Line, Simple Order and Group Assignment seemed really interesting for us. Indeed, they all consist of placing elements in the right position.

For the first one, the particularity is that you have to place them on a line with scaled units (for instance it can be years and you have to place historical events in the corresponding period of time) as can be seen in figure 2.1.

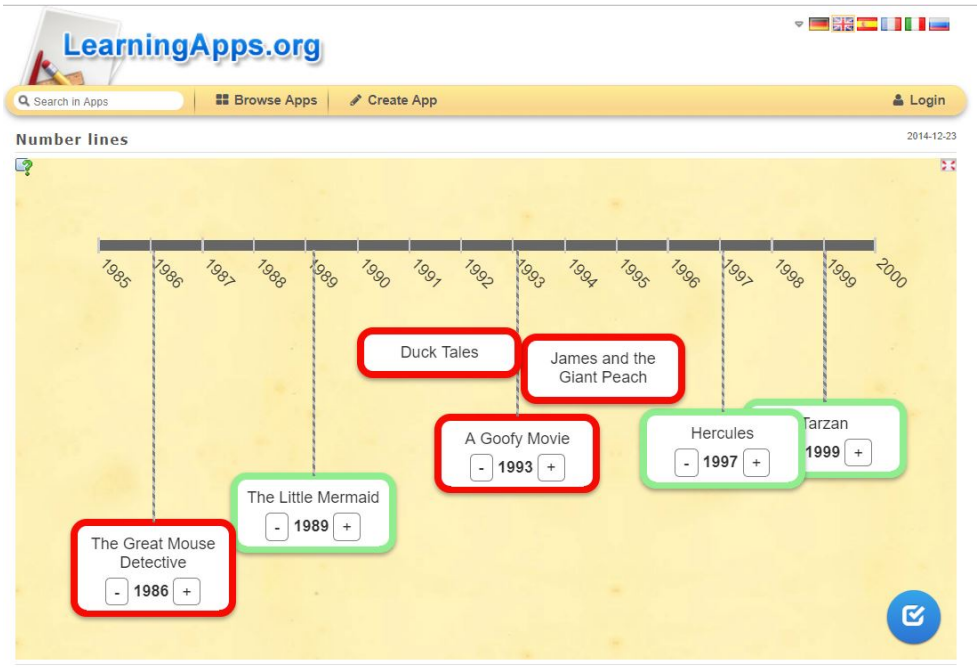



Figure 2.1: Number Line

The second one, consists in placing elements in the right corresponding set as illustrated figure 2.2. For example, if you take an example where the sets are fruits and vegetables, you would put the apple in the first set.


Figure 2.2: Group Assignment

For the third one, the elements can be moved freely on a canvas and the app will automatically establish their order as seen in figure 2.3 and 2.4.

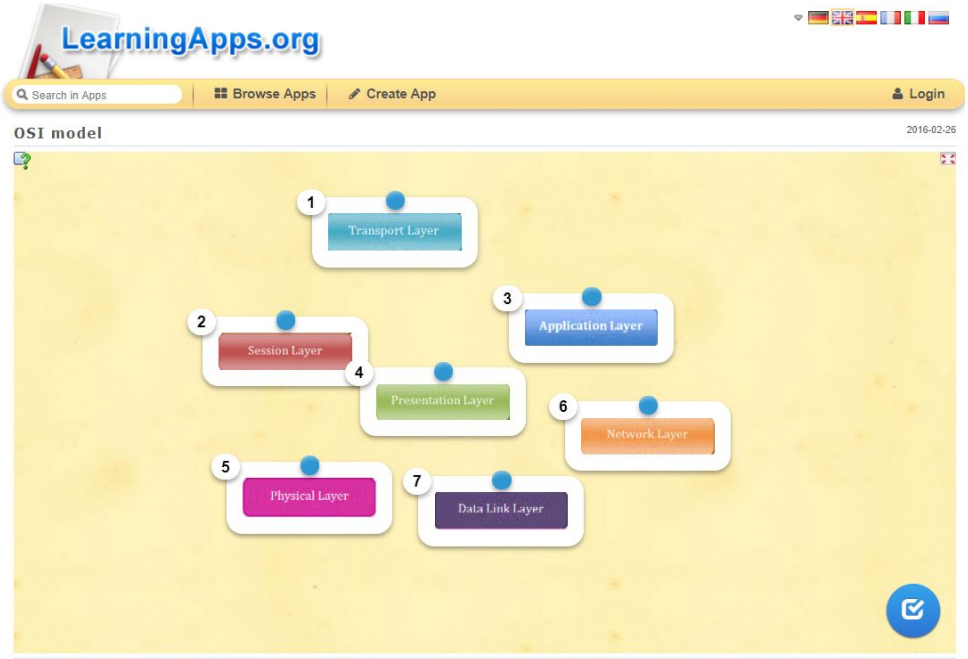



Figure 2.3: Simple Order


Figure 2.4: Simple Order resolved

The Oscar team asked us to take this website (especially the three exercises that we talked about) as an example of what they would like the module to be.

## 2.2 Adobe captivate

The famous software company *Adobe System* is developing *Adobe Captivate*, a tool allowing the creation of drag & drop based exercises. For example, the figure 2.5 shows an exercise where the user has to replace the name of elements of the hearts at the corresponding place.

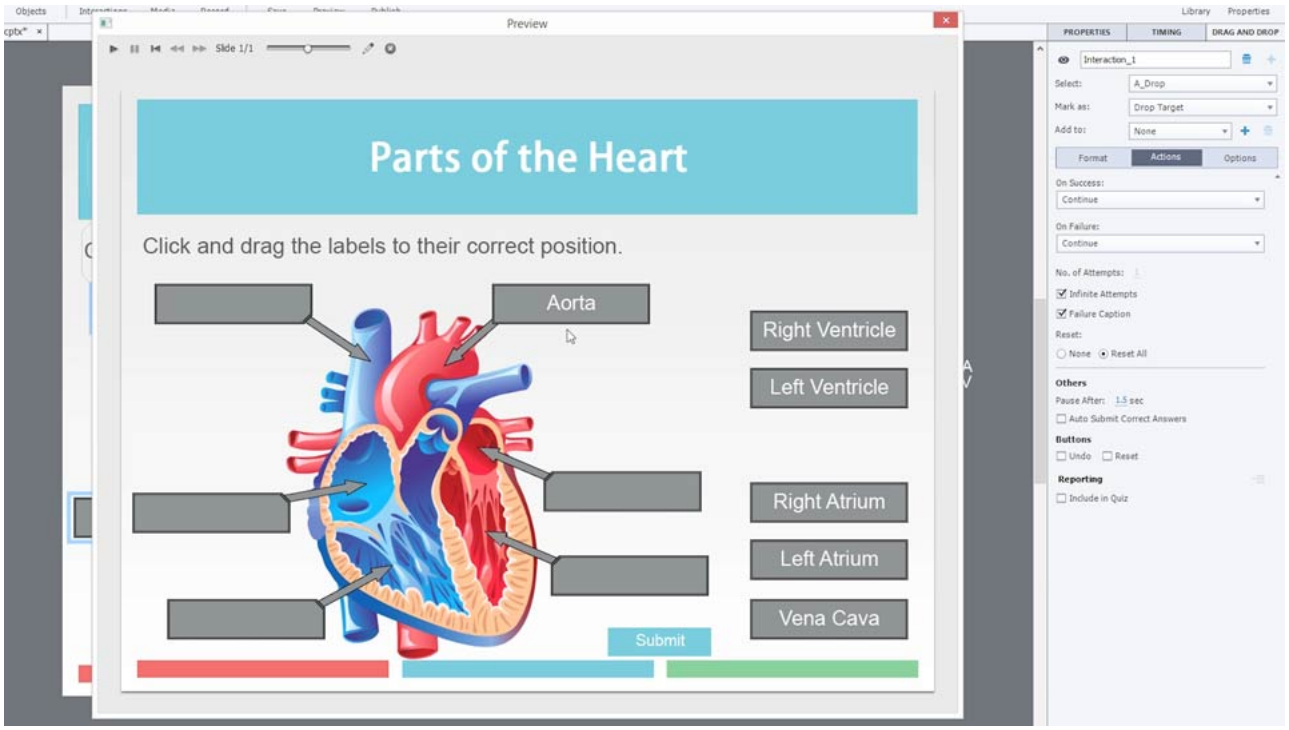



Figure 2.5: Creation of an interactive exercise through Adobe captivate [9]

## 2.3 Language learning

Placing words in the right order is also a common exercise in order to learn a new language. There are several applications that are doing this such as *Tolearnenglish* [10] or *Duolingo* [11].
Tolearnenglish is a website which, as stated in the name, helps to learn english. Among the exercises proposed, one of them displays elements of a sentence in a random order, and you have to reorder them. However, this site doesn't use a drag and drop tool, you have to click on elements so they disappear and reappear in a box next to their location. So of course you have to click on the first word first, then the second, ... Moreover, if you want to correct something, you have to click on the window and have to restart all over, which is not very intuitive. A screenshot can be seen figure 2.6.

Duolingo is a very well-known language learning application with a web interface and a smartphone application. Basically it uses the same principles as tolearnenglish, except that for the mobile application you have to use your finger. For the web application, you can use drag and drop elements using the mouse. A screenshot of this app is shown figure 2.7.

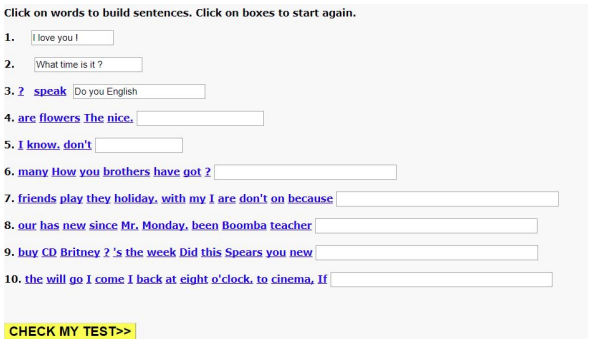



Figure 2.6: ToLearnEnglish


Figure 2.7: Duolingo

# Chapter 3

# Requirements analysis

Throughout this chapter, we will analyze the requirements on multiple angles that will allow us to extract different details on the desired module.

As the module is internally divided in two parts, the creation of questions by the teacher and the answering by the students, each approached angle of the analysis will be divided into the teacher and student's point of view.

## 3.1 Client Requirements

For its website, the Oscar team, represented by Laurent Fourny (the founder), wanted a new way for teachers to evaluate the level of their students. This new module should allow teachers to create exercises capable of evaluating the mathematical reasoning skills of students. Those exercises would consist of different elements that the student should place according to the teacher's intended order.

For example, let's think of the steps needed to draw the "incircle" of a triangle. In order to do that, the student has to go through a number of steps such as drawing the bisectrix, the circle, ... Those steps can be decomposed into smaller steps, such as finding the points that the bisectrix goes through. Therefore the teacher could create one "element" for each step with his description inside. He would also provide the order of the elements corresponding to the "right" one. Then the student would be able to "do" the exercise, more precisely he would be given the different steps in a random order and he would have to place them in the correct disposition.

In order to be usable by real students and teachers, the module needed to be user-friendly. That's why the Oscar team requested a drag&drop interface, where the student could move the elements with his mouse, which is more intuitive than clicking or re-writing blocks for example.

## 3.2 User Stories

In order to better understand the different tasks that the module should allow to accomplish, we wrote user stories. Each user story is a sentence that defines a functionality to be developed by answering three questions, "Who ?" , "What ?" , and "Why ?".

### Teacher's point of view

A. As a teacher, I can create a new drag and drop exercise to assess my student.

B. As a teacher, I can add as many blocks as wished to complete my exercise.

C. As a teacher, I can fill the different blocks with either a text or an image to complete my exercise.

D. As a teacher, I can move the elements of answer how I want across a canvas so I can give them the right order.

E. As a teacher, I can choose to make either an ordering or a classification exercise so it's the most appropriate for my question.

F. As a teacher, I can validate my new question and make it accessible for students so my students can be assessed on this subject.

### Student's point of view

A. As a student, I can access a question of reasoning and read the statement so I can have recommendations corresponding to my results once I finished it.

B. As a student, I can move the elements how I want across the canvas so I can order them and answer the question.

C. As a student, if in training mode. I can validate my answer to get appropriate feedback.

D. As a student, if in exam mode, I get recommendations for appropriate course material to extend my knowledge.

Unfortunately, there is no training mode available in the versions of Oscar we worked with. Consequently, for the last two user stories, we will implement the basis for it but it will not be available until a training mode is implemented.

## 3.3 User Scenarios

In order to better understand the behavior of the module, we wrote the following user scenarios. Each user scenario is a fictitious story of a user using the module with the previously defined goals.

### Teacher's point of view

Robert, a registered teacher on the website, is creating a question for his students :

1. Robert selects the question type "ordering question".
2. Robert writes the instruction of his question.
3. Robert defines the blocks to be ordered that will be available for the students.
4. Robert puts the blocks in the order that he defines as the right one.
5. Robert validates and submits his exercise.

### Student's point of view

Philip, a student from secondary school, is answering a reasoning question created by Robert, his teacher.

1. Philip reads the statement of the question.
2. Philip moves the elements at his will on the provided drag and drop section.
3. Philip validates his answer.
4. Philip receives recommendations for appropriate course material if in exam mode.

## 3.4 Mock-ups

In order to have a more visual idea of the module that we would implement, we have designed some mock-ups shown in the following subsections. They allowed us to be sure that we understood well the desiderata of the client. Of course, it was only an idea of what the module could look like. Progressively, as we were implementing it, we adapted our ideas thanks to the fact that we could see what it looked like in reality and based on the feedback of the client.

### Teacher's point of view

As the teacher needs to be able to create a complete new question, our first idea was to have a basic form to be filled up, as can be seen figure 3.1. The teacher would first have to choose between the different types of canvas and then to add the desired boxes and their information. At the end, the teacher would submit the exercise reviewing a complete preview resulting of the filled form before confirming the whole process of creation.


Figure 3.1: Our first idea for an interface for the creation of a question

However, our supervisor had the idea of having a "what you see is what you get" interface, i.e the teacher would see the same things that will be displayed to the students. So we decided that the teacher would first have to choose the type of question he wants, and then the corresponding canvas would be displayed. Then he could just add/delete blocks, and finally arrange them in the correct order. Eventually he would submit the whole exercise and we would just retrieve all the information to be stored in database. The result of this idea can be seen figure 3.2

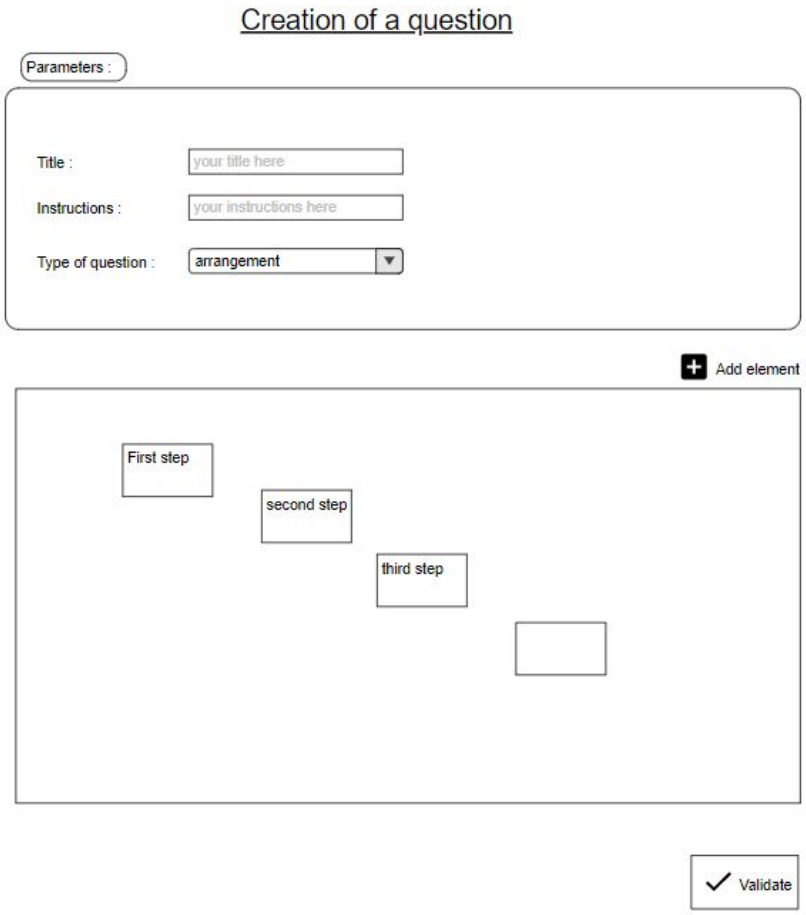
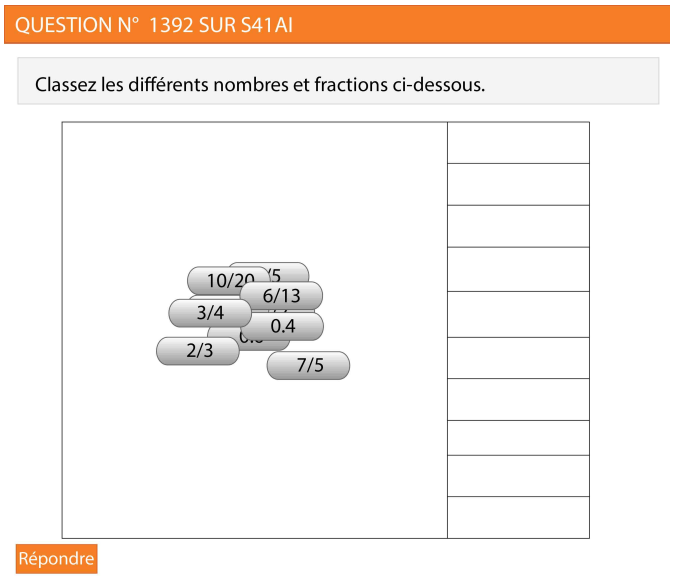



Figure 3.2: Our second idea for an interface for the creation of a question

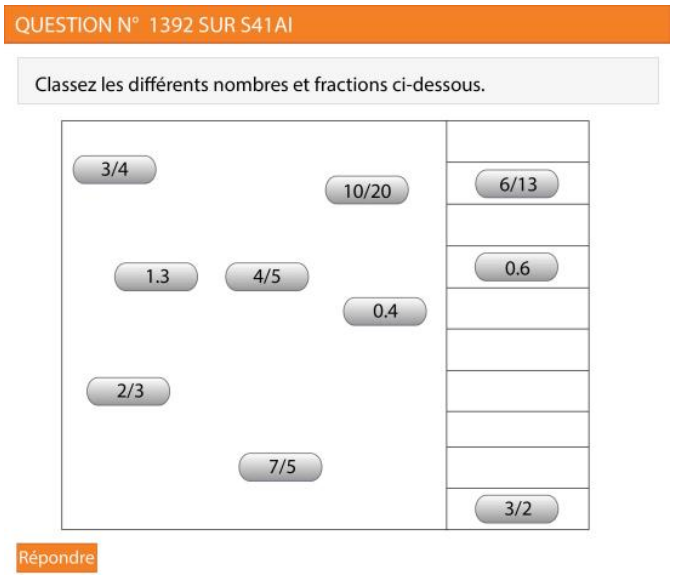
### Student's point of view

As for the student, he will only need to receive the full already established exercise in which he will not be able to modify any information. But the student will need to be able to move the different boxes to form the suitable answer to the exercise.



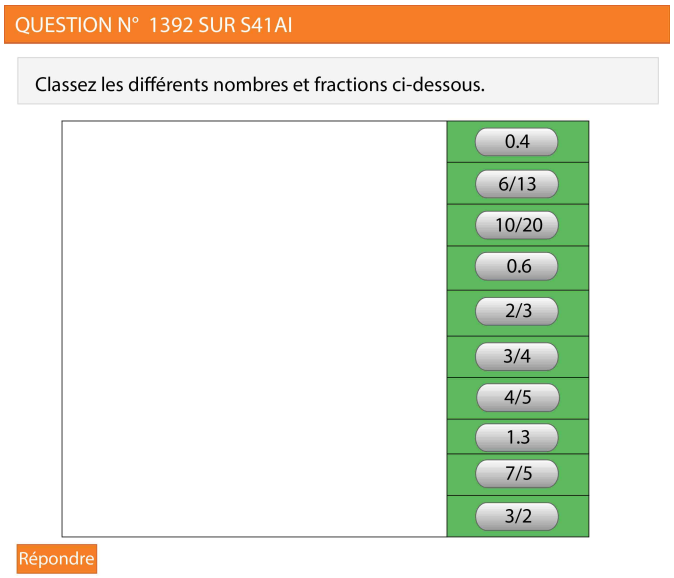

The first mock-up shows the initial display of the question visible by the student. The elements are shuffled in the middle and each element isn't necessarily visible.

Figure 3.3: Mock-up : ordering exercise resolving




Then the student can sort and move the elements to the right column.

Figure 3.4: Mock-up : ordering exercise resolving




When the student has placed every answer to the chosen place. He can validate his answer.

Figure 3.5: Mock-up : ordering exercise resolving


If it's a training, the good answers will be represented by green and wrong answers by red. If it's a test, the student will be redirected to his home page with appropriate course material recommendations according to the result of the test

Figure 3.6: Mock-up : ordering exercise resolving

As we were implementing our module, both us and the client thought that it would be more handy if the user could place the boxes anywhere in the canvas rather than in little emplacements. The order would be calculated with their absolute position, with the origin point being at the top left of the canvas. The result of this idea can be seen in figure 4.10.

Moreover, the last two mock-ups correspond to the training mode, which is not implemented in Oscar yet. As a consequence, when the student validates his answer, he will not have a correction. Instead, he will be redirected to his home page, with his recommendations updated according to his results (we'll talk more about it in the implementation section).

## 3.5 Functionalities

Here is an exhaustive list of functionalities that needs to be implemented.

### 3.5.1 Draggable Blocks

The core idea of the module is the "drag and drop" interface. Our module uses JQuery (we will talk about it later) in order to enable this functionality. The idea of "drag and drop" here is to have elements that can be moved across a pre-defined container by clicking on them and holding the click. While the click is held, the element will simply follow the mouse, and then the user can release the click when the element is in the wanted position.

### 3.5.2 Content

In order to be relevant, the boxes must have content. The first content we need to implement is simple text that the teacher would have to type with his keyboard. Then, as the teacher will write mathematic questions, it's very important to be able to display well written mathematic functions.

Additionally, we thought of one more sophisticated content, images. An image content would give a lot of flexibility to the possible content of the boxes. Instead of drawing graphs directly into the website, they could be drawn in another application or on a real board and added through this functionality.

### 3.5.3 Three types of canvas

As the exercises' variety in mathematics is vast, we will need to implement multiple types of questions based on the same environment (drag and drop) but with different goals. The client required the three following types of exercises : ordering, classification and what we called "graduated line".

#### Ordering

The first basic question type required by the client was to allow the creation of ordering exercises. For example, the statement could be : "Given a certain set of fractions, order those numbers in ascending order".

With drag and drop, the most natural way to do it from the student point of view is to display the elements in random positions and then letting them arrange the blocks by dragging them. When he has finished and submitted his answer, we then need to retrieve the order using the positions of the boxes.

#### Classification

The next type of exercise was the classification exercise. For example, "Given a certain set of numbers, classify them in the appropriate sets". (figure 3.7)

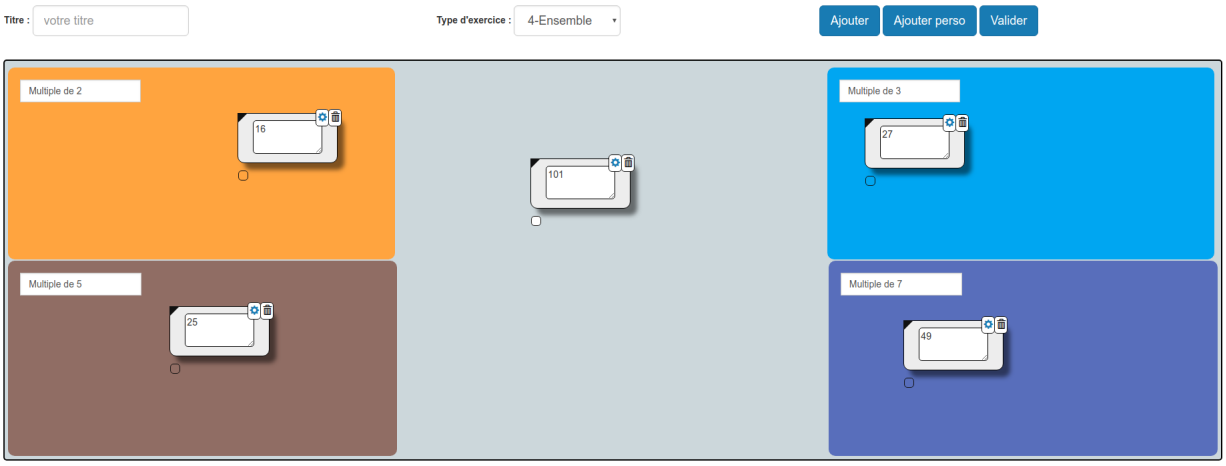



Figure 3.7: Example of classification exercise

With a given number of sets, each with different properties, the student is tasked to put each element in the correct set. The teacher should be able to choose the properties of the sets and their title.

#### Graduated line

The last type of exercise of our module is what we called "graduated line". In this type of exercise, the goal of the student is to put elements on an axis in the right interval, just as in the "number line" exercise from Learning Apps.

To have a better idea of how it works, let's think of the following example of the creation of such exercise. First, the teacher has to specify an interval (for example 0 to 3) and then a number, for example 6. In this case, a line will be displayed and will have 7 indexes (0, 0.5, 1, 1.5 until 3.0). Then the teacher can create boxes with for example fractions within the range of these intervals (11/6, 27/10, ...). Eventually the student will have to replace the boxes in their corresponding interval.

### 3.5.4 Feedback

#### Assessment mode

In assessment mode, once the student submitted his answers, there are two possibilities : either he succeeds, and in this case the skill is validated, or he fails, and in this case the skill is not validated. In addition, Oscar will suggest the student appropriate resources so he can improve himself.

#### Training mode

In training mode, the student needs to have a possibility to learn from his mistakes. And in the same way, the teacher needs a way to give appropriate feedback to the student.

There are two possible types of feedback :

1. General feedback on the question. This feedback would give hints to how to approach the question type and its subject.
2. Specialized feedback on an answer. This feedback would use the answer of the student and target what were his mistakes.

However, as we previously said, the training mode was not available in the version of the website we worked on. Therefore this functionality will be implemented in a basic way for further developments but will not be functional.

# Chapter 4

# Implementation

In this section we talk about the implementation of our work (which technologies we used, how we did it, ...) .

It is important to note that for reasons beyond our control, the site being in development, we had access to Oscar's code much later than we expected. As a result, we had to begin coding before having access to the code and therefore, we had to do sometimes the work twice since it was not the same implementation to make our code work in a stand-alone fashion or integrated in the website. We thus had less time than foreseen for the development.

## 4.1 Approach

### 4.1.1 Methodology

We used an "agile way of thinking" to implement our module. Even with a team of two developers only, it appeared like the best organization for our project.

"Agile software development describes an approach to software development under which requirements and solutions evolve through the collaborative effort of self-organizing and cross-functional teams and their customer(s)/end users(s). It advocates adaptive planning, evolutionary development, early delivery, and continual improvement, and it encourages rapid and flexible response to change." (Wikipedia) [13]

This methodology defines roles for each member of the project, such as the product owner, represented by Laurent Fourny here. He also played the role of stakeholder since he represented himself. There are also the team members, us (Nicolas Tonon and Antoine Habran). Finally, Kim Mens was somehow playing the role of team leader, even though he was not doing every aspects of this role since it was not "his" project, he was kind of an advisor.

Agile also cuts the implementation phase into small increments. This way the up-front planning is minimized. Each iteration (or sprints) are short time frames (usually one or two weeks) in which precise functionalities have to be planned, analyzed, coded and tested.

Each sprint resulted in a meeting with the client where he would give us a feedback on the implemented functionalities and where we would define the following things to do.

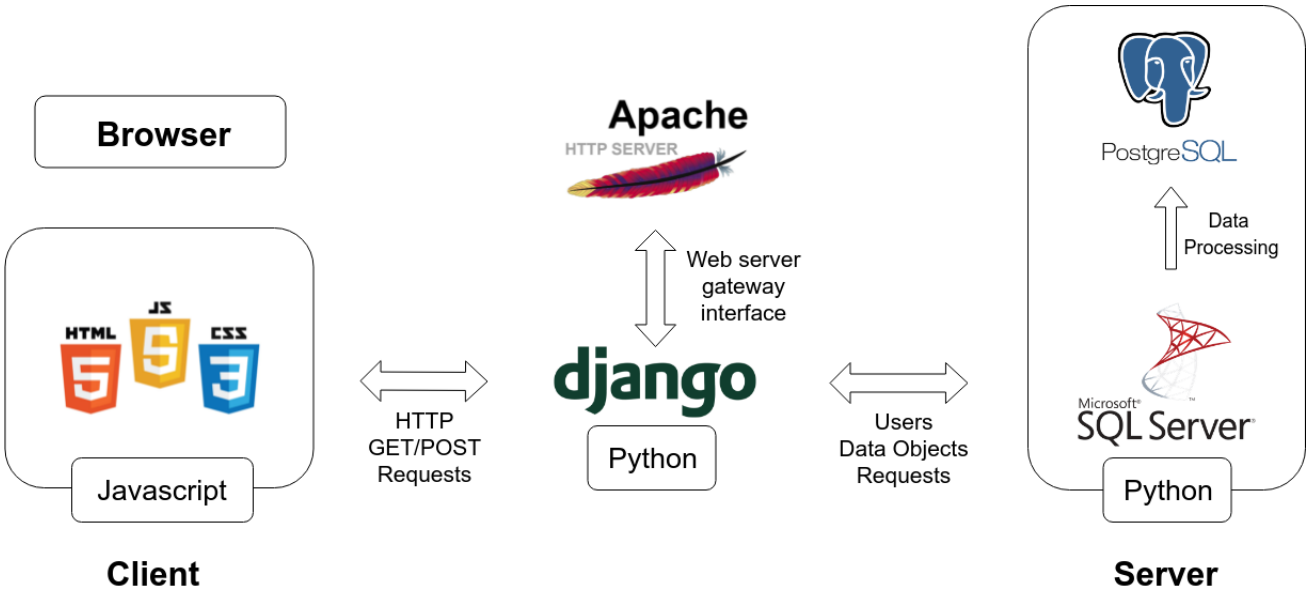



Figure 4.1: The functioning of Django in a web application [16]

### 4.1.2 Philosophy

Agile was really adapted to us since, as the name suggest, it implies an agile behavior. By this, we mean that as we worked on the project, new ideas came to us and through discussion with the client, we took the initial idea and made it evolve to what we hope is a better result.

We were also agile in our way of working. We didn't strictly divide the work from the beginning, we tried both to work on every aspect of the module, from the analysis phase up to the integration. Through communication we regularly defined which part of the work we wanted to do, and thanks to that, we were really in phase with the agile philosophy

## 4.2 Framework

Oscar is developed with the web framework Django. A web framework is a tool designed to support the development of web applications. It provides standardization in the way the development is done, it aims to automate tasks in order to prevent developers to redo work that has been done before. To be more precise, many websites require similar functionalities. For example an account system (for signing in and signing up), an administration panel or even a file management system [14].

Therefore, rather than having to re-implement those things from scratch every time, the framework automates such functions. For example, a web framework can facilitate the interactions with the databases by providing a library with functions that are easy to use for the developer but that do the whole process of interaction with the database without showing it [14]. That's exactly one of the features of Django.

### 4.2.1 Django

Django is a free and Open-Source web framework, written in Python which follows the model-view-template or MVT (that will be explained later) [15]. As said in the previous section, the goal of Django is to provide a set of tools helping to develop web applications more easily and quickly. A representation of how Django works can be seen figure 4.1.

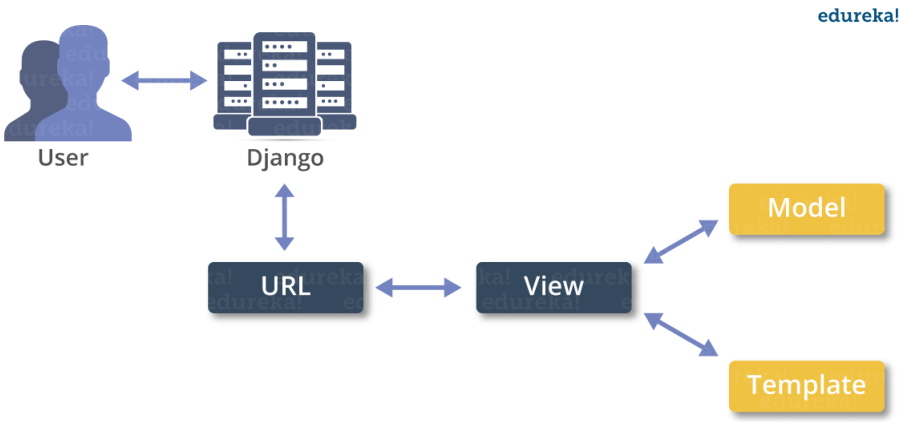



Figure 4.2: Model view template representation [17]

**Model View Template**

Django is based on the MVT (model-view-template) architecture. The MVT is very close to the best known MVC architecture (model-view-controller). "The model-view-controller is an architectural pattern commonly used for developing user interfaces that divides an application into three interconnected parts. This is done to separate internal representations of information from the ways information is presented to and accepted from the user. The MVC design pattern decouples these major components allowing for efficient code reuse and parallel development." [18]

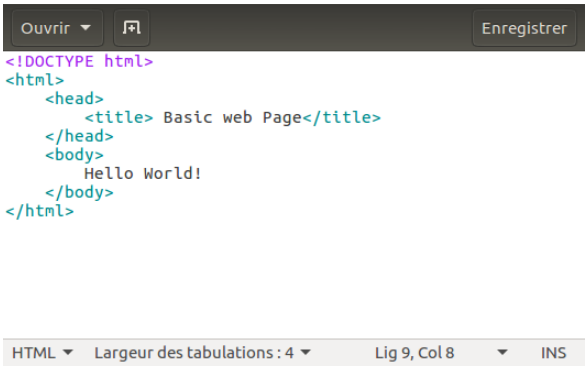
The three parts are the following [19] :

- model : An abstraction of the data. It's not the data itself, but a representation of it. It allow to have transactions with the data without knowing the intricacies of the underlying database through an interface.
- view : The way the data are represented on the screen. The view also provides an interface to collect the user inputs.
- controller : It controls what is exchanged between the model and the view. More precisely, it controls what can be pulled from the database and passed to the view so it can be displayed to the user. The controller also get the inputs of the user through the view and passes it to the model.

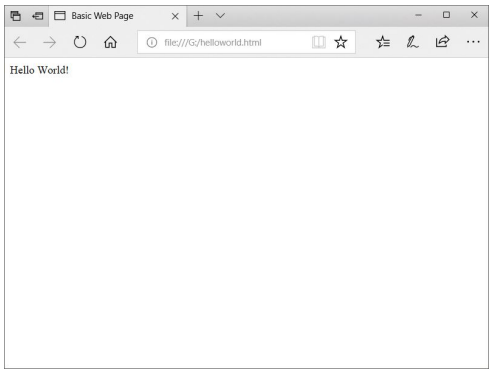
The main objectives of the MVC architecture are to ease both simultaneous development and code reuse. Django is essentially based on the MVC but with a different way of handling the controller part : it is handled by the framework itself, hence the "template" rather than the "controller". A schematic representation of the functioning of the MVT can be seen figure 4.2.

## 4.3 Front-End

Our module uses a combination of HAML, CSS and Javascript for the front-end (i.e the look and behavior of the pages). We chose them because they are used in Oscar. Moreover, those technologies (if we consider HAML as HTML, we will explain why later) are the most commonly used technologies for the creation of web pages. They all have a different role. HTML handles the structure and gives sense to the content, CSS manages the look and finally JavaScript is in charge of the dynamic behavior.

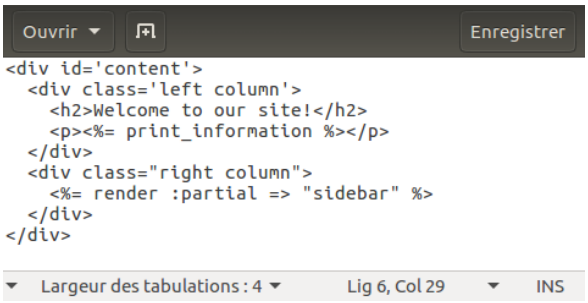



(a) HTML code

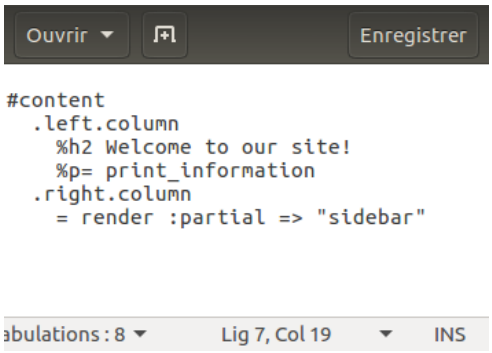



(b) Result

Figure 4.3: Example of a basic form in HTML without BootStrap




(a) HTML code




(b) Equivalent HAML code

Figure 4.4: Example of a basic HTML code and its HAML equivalent [21]

### 4.3.1 HTML

"HTML is the standard markup language for creating web pages." It describes the structure of Web pages using markup, its elements (represented by tags such as "heading" or "paragraph") are the building blocks of HTML pages [20].

An example of a trivial HTML code can be seen on figure 4.3. HTML is often used with CSS (Cascading Style Sheets) for the aspect of the elements of the page such as the color, the size or even the place of those elements, and JS (JavaScript) for its behavior.

### 4.3.2 HAML

"HTML Abstraction Markup Language" or HAML is a simplified version (mainly in term of syntax) of HTML. The figure 4.4 shows a simple HTML file and its equivalent in HAML. We can see that the HAML code is much more concise and simpler to read.

In our module, we use HAML to organize the interface of the teachers and the students. The figure 4.5 shows how we organize the drop-down menu and the buttons that add boxes. The result can be seen figure 4.12.

### 4.3.3 CSS

"Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colors, and fonts. This separation can improve content

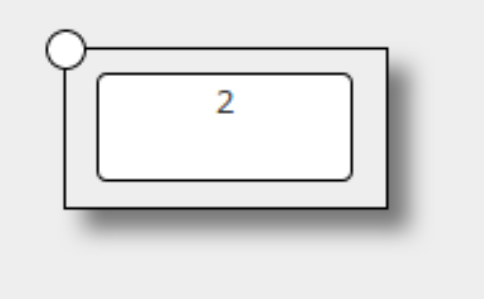
```
.form-group
  %label.control-label.col-sm-2 Type de canvas
  .col-sm-10
    %select.form-control{ng-model: "question.canva.type", ng-required: "true", ng-change: "changeCanva(question)", id: "selectCanva"}
      %option{value: ""}
      %option{value: "ranking"} Classement
      %option{value: "2-set"} 2-Ensemble
      %option{value: "4-set"} 4-Ensemble
      %option{value: "graduatedLine"} Ligne graduée

#AddBlock.form-group{style:"display: none;"}
  .col-sm-12
    %p.help-block Pour ajouter une boite dans le canva, il vous suffit d'appuyer sur un des bouttons ci-dessous dépendemment du type de boites que vous souhaitez.
    Une boite dans le canvas est déplaçable, redimensionable et personnalisable.
    %p.help-block En haut à gauche d'une boite se trouve un indicateur d'ordre ou de placement adapté au type de canvas. Et, en haut à droite,
    se trouve un boutton permettant de supprimer la boite et un autre permettant de personnaliser le contenu de celle-ci.
  .col-sm-10
    %button.btn.btn-success.add-question{ng-click: "createDraggableBlock(0, question, topIndex)", type: "button", id:"addBlockText", style:"display: inline-block;"}
      %span.glyphicon.glyphicon-plus
      Text
    %button.btn.btn-success.add-question{ng-click: "createDraggableBlock(1, question, topIndex)", type: "button", id:"addBlockMath", style:"display: inline-block;"}
      %span.glyphicon.glyphicon-plus
      Math
    %button.btn.btn-success.add-question{ng-click: "createDraggableBlock(2, question, topIndex)", type: "button", id:"addBlockFile", style:"display: inline-block;"}
      %span.glyphicon.glyphicon-plus
      Image
    %button.btn.btn-success.add-question{ng-click: "openBlockInfoDisplay(-1,question, topIndex)", type: "button", id:"addBlockGeneral", style:"display: inline-block;"}
      %span.glyphicon.glyphicon-plus
      Perso
```

Figure 4.5: HAML code for the teacher page

```
.dnd-draggable {
  width: 160px; min-width: 160px; max-width: 500px;
  height: 80px; min-height: 80px; max-height: 500px;
  float: left;
  text-align: center;
  margin: 0 10px 10px 0;
  box-shadow: 10px 10px 10px rgba(0, 0, 0, 0.5);
  background-color: #EEEEEE;
  border:2px solid black;;
  cursor: move;
}
```

(a) css code




(b) result

Figure 4.6: Example of a css code in our module

accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content." (wikipedia) [22]

CSS allows a more efficient and easier control of the look of the content written in HTML.

The figure 4.6 shows an example of the use of css in our code. Here, we configure a box with many parameters among which the size, the height and width or even the background color.

### 4.3.4 JavaScript

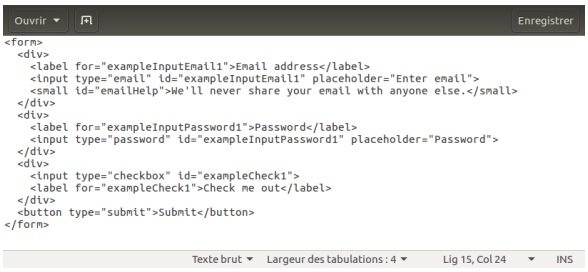
"JavaScript is a full-fledged programming language interpreter embedded inside a web browser. It does anything that a regular language like Java allows, among which :

- Declare variables
- Store and retrieve values
- Define and invoke functions
- Define your own classes" [23]

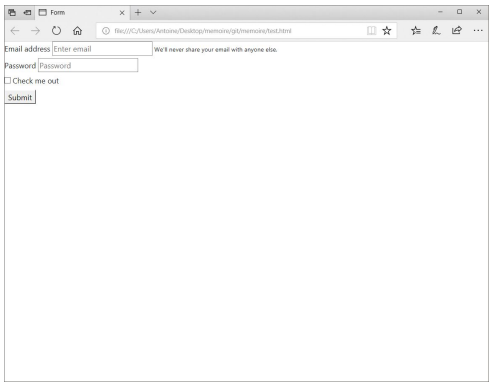
It can also interact with the HTML/CSS of the pages : adding, deleting or modifying content , ... Usually, JavaScript is used to have dynamic behavior in a web page, for example, it can change HTML content

```
function order() {
  if(canva == "ranking") {
    tab.sort(function(a,b){return (a.position().left-b.position().left)+(a.position().top-b.position().top);});
    for (var i = 0; i < num; i++) {
      var number = tab[i].attr('id').slice(9);
      $("#order"+number).html(i);
      $("#dnd-"+numQuest+"-"+number).val(i);
    }
  }
  else if(canva=="2-set" || canva=="4-set") {
    for (var i = 0; i < num; i++) {
      var elementid = tab[i].attr('id').slice(9);
      $("#dnd-"+numQuest+"-"+elementid).val($("#draggable"+number).attr('set'));
    }
  }
}
```

Figure 4.7: JavaScript code for actualizing the rank of blocks.




(a) HTML code



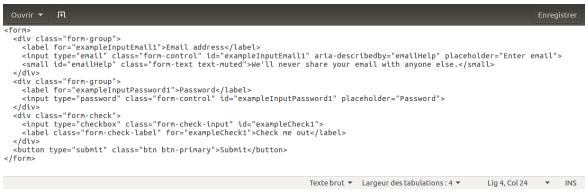

(b) Result

Figure 4.8: Example of a basic form in HTML without BootStrap

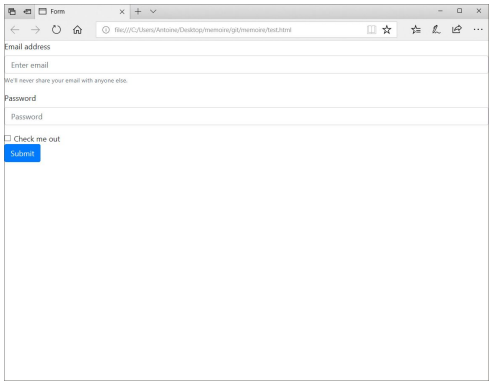
after a specific event occurred. In our module, we use JavaScript in order to refresh the state of the blocks. More precisely, if we are in an ordering exercise, we have to determine the rank of each block in real time. So when a user drags a block, its rank will be modified according to the new position of each block. The code for this functionality is shown figure 4.7

### 4.3.5 Libraries

Since CSS and JS are somehow pretty "low-level", using them from scratch can be a waste of time. Indeed, there are libraries for those technologies that allow us to build things more quickly and easily. For example, when one makes a basic form in HTML, it looks stern at first, and modifying it with basic CSS would take unnecessary time.

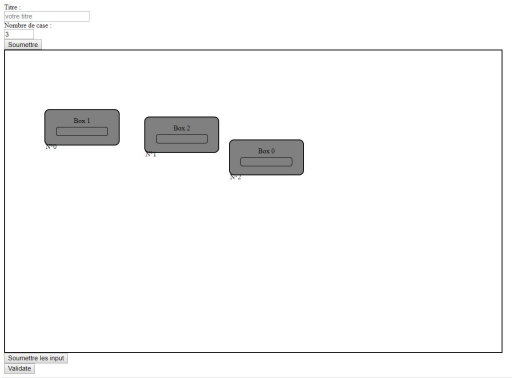



(a) HTML code

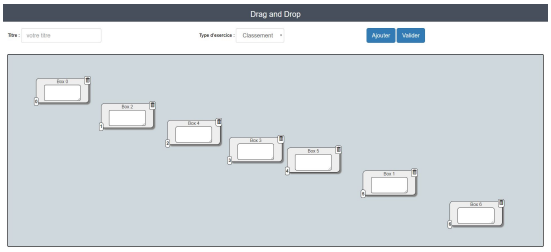



(b) Result

Figure 4.9: Example of a basic form in HTML with BootStrap




(a) Before Bootstrap




(b) After Bootstrap

Figure 4.10: Enhancing our module with BootStrap

### BootStrap

Thanks to a library like BootStrap [12], you can save a lot of time by taking the already existing element like buttons, input fields, ... An example of the use of such a library can be seen figure 4.8 where you can see an example of a code for a basic form and the result in a browser, and figure 4.9 where you can see the same code but with using Bootstrap [24]. You can also see that the number of line of code is the same in both cases. Its simplicity and beauty was perfect for our needs since we wanted something efficient and good looking. The figures 4.10a and 4.10b show before and after the use of BootStrap on the first versions of our module. We can clearly see an enhancement both visually and ergonomically.

### Drag&Drop

Since our module is based mostly on "Drag&Drop", we needed to find a library implementing it and which would also be flexible enough so we can have a perfectly fit result. On a previous project done with Oscar for the class of software development project, we used JQuery [25], a powerful JavaScript library implementing amongst other things a simple but efficient Drag&Drop functionality that the Beta version of Oscar used. It should be noted that halfway through the implementation, the client announced us that the new version of Oscar instead would use React [26], a JavaScript library for building user interfaces.

After discussion, we decided to continue with JQuery because, firstly, we wouldn't have had the time to start over the module with a library that we would have to learn from scratch and secondly, the site not being fully functional yet with React, the integration would have been even more complicated than with the current version.

### MathQuill

In order to display mathematical writing, we chose MathQuill a library used by Oscar that we also used during our previous project done with Oscar. MathQuill is a web formula editor designed to make typing math easy and beautiful. It transforms Latex into displayable text. However, as a result of an update, MathQuill is not working properly anymore, thus removing the functionality of creating mathematical function in latex that would have been displayed into the boxes composing a mathematical reasoning exercise.

Furthermore, we learned during the project from the Oscar team that it was one the reasons for which they were switching to the React framework.

## 4.4 Back-end

The back-end of Oscar being completely handled by Django. We didn't had to implement any big function in the back-end. However, we had to tweak some functions to accept our new question type and to enable the evaluation and withdrawal of the student answers.

## 4.5 The module and its Integration

In this section, we will talk about the proper implementation and integration of the module. The code is available at https://github.com/ntonon/oscareducation.

As we integrated our work into the Oscar's website, all of the code is not ours. A way to see what is from us is to check the commits since April. Also, here is the exhaustive list of all the files we worked on :

1. */templates/examinations/exercice_rendering.haml* were modified in order to display the new question type for the students.
2. */templates/professor/exercice/validation_form.haml* and */templates/examinations/drag_and_drop.haml* were modified in order to display the elements to allow the creation of a new reasoning assessment question for the teachers.
3. */promotions/static/js/drag_and_drop.js* was created to handle the javascript for a student to answer a question of our type.
4. */promotions/static/js/professor_validate_exercice.js* was modified to gather the different informations of the created questions of teachers.
5. */promotions/models.py*, *examinations/validate.py* and *student/views.py* were modified to accept our new type of question at the validation, correction and evaluation.
6. *oscar/static/css/drag_and_drop.css* was created to modify the visual displays of the different templates.
7. *templates/base.haml* was modified to link the added css in the project.

If you desire to test it, you must first install the Oscar website by following the instructions on the "readme". Then, you have to be connected as a teacher, choose a class, and then create an online test.

### 4.5.1 Oscar

Before digging in the integration, let's first explain a bit how Oscar works and where our module will be integrated.

In Oscar, teachers have classes composed of students. One of the functionality is the ability for the teachers to propose resources for specific topics. He can also propose tests concerning those topics. Beside that, the students can access the resources and the tests proposed by their teachers. And once he passes a test, if he fails, the site will suggest resources related to the topic. Our goal is to create a new module proposing the types of exercises mentioned in the requirement section for the creation of tests.

### 4.5.2 Teacher's side

We began the integration by adding our module to the questions creation section inside the test creation module for teachers. So when a new test is created, while creating a question, a new type, "question de raisonnement mathétmatique", now appears as we can see figure 4.11.

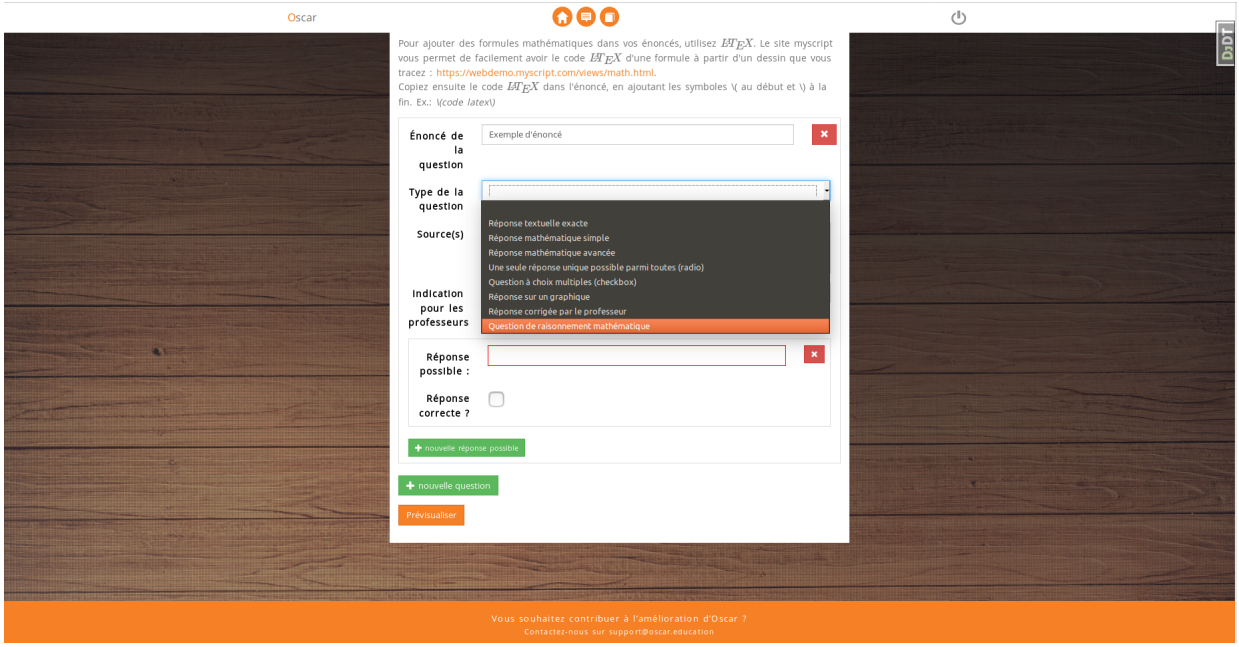



Figure 4.11: Creation of a new question

After the teacher chose the new question type, he can choose the canvas of his exercise. On figure 4.12, we can see an example with a 2-Set canvas for the creation of a classification exercise while the figure 4.13 shows the creation of a ranking exercise. Finally, the figure 4.14 shows the creation of a graduated line exercise.

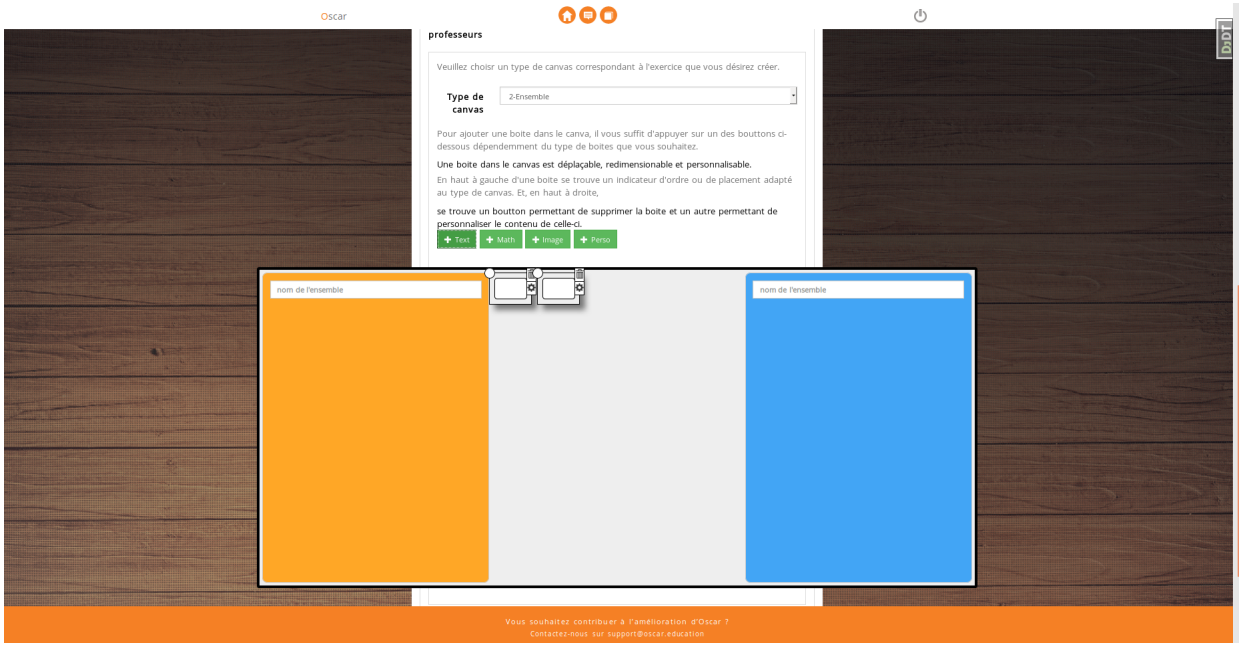



Figure 4.12: Creation of a 2-Set question

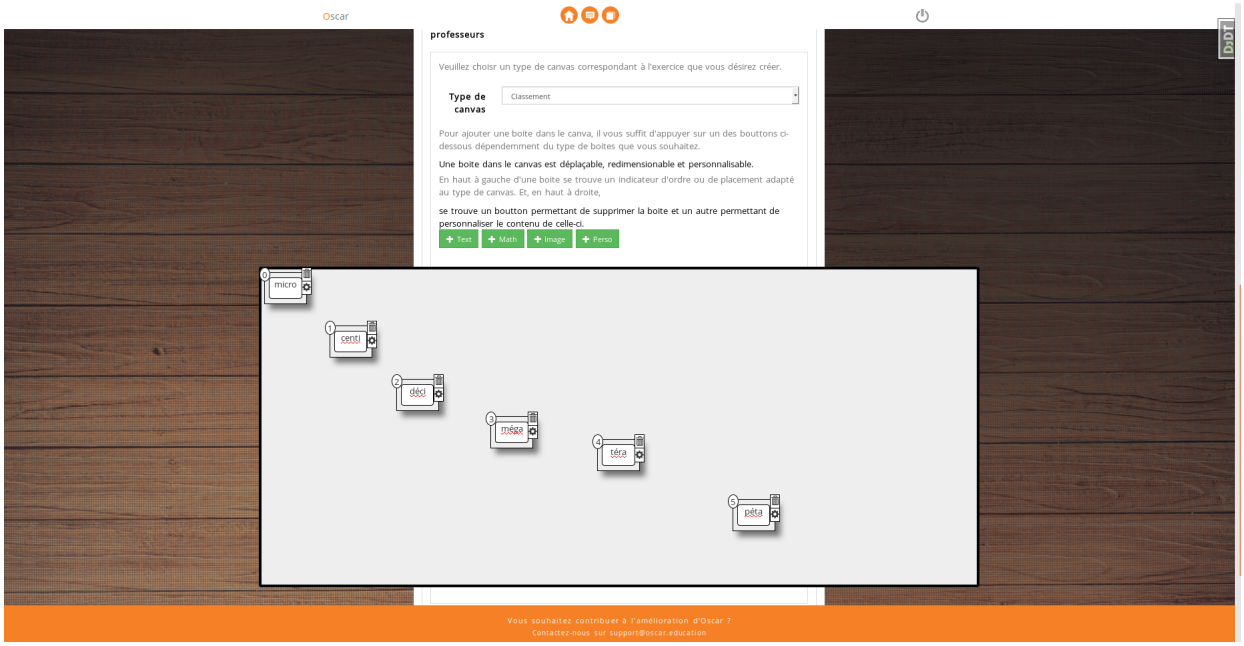



Figure 4.13: Creation of a ranking question

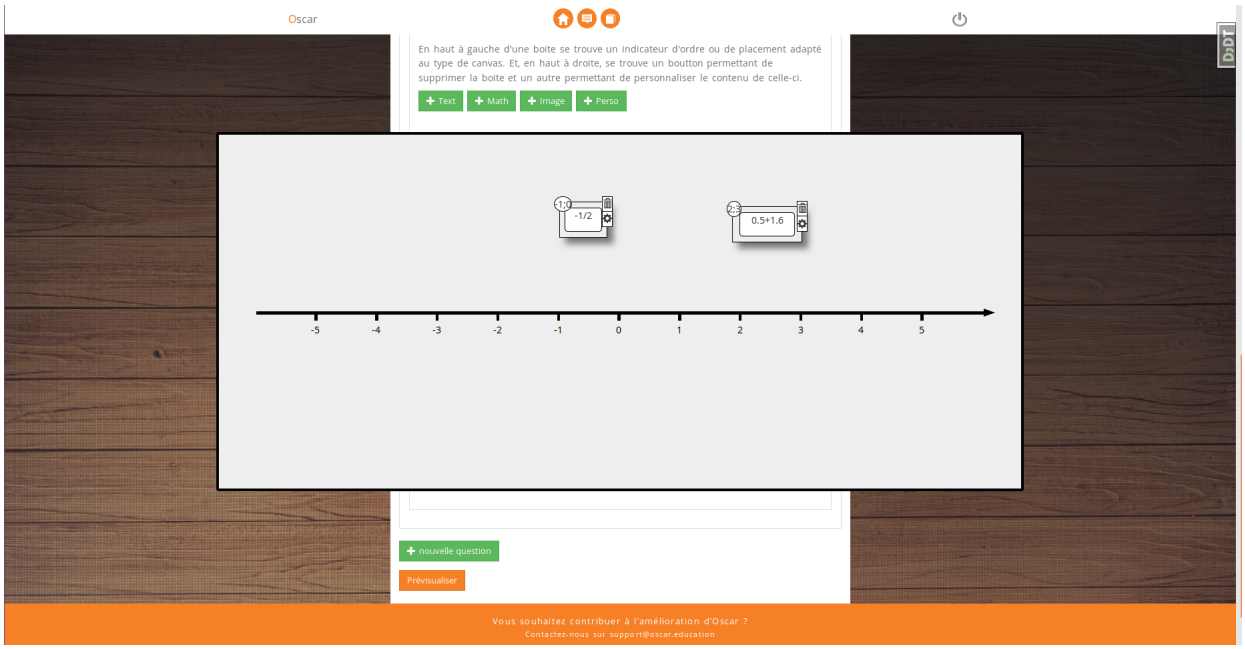



Figure 4.14: Creation of a graduated line exercise

Once the teacher chose the type of the canvas, he can start adding his blocks. As mentioned in the requirements, we implemented the following types of content for the blocks : text, images and math. For each of these contents, there is a corresponding add button. We also have a fourth button, that we called "perso". It allow to configure your button from scratch through a pop-up which we will talk about later. Moreover, different types of blocks can be putted together in a question. For instance, let's think of a teacher doing set exercises where the student has to classify shapes. The teacher could give either an image of a shape with info on it or a textual definition of it.

Once a block is added, it has to be completed. For the text, the teacher has to type it with the keyboard. Concerning the images, a button is placed at the bottom of the box as can be seen figure 4.15. This button opens a window allowing the user to upload a picture from his computer. Once the image is uploaded, it appears in the block. Since the image's size will be adapted in order to fit in the block, it is likely that the image will be too small to be seen correctly, so we implemented a zoom function : when the image is clicked, it will appear bigger as can be seen figure 4.16.

Moreover, each block has two buttons. The first one, with a trash icon, deletes the blocks when clicked. The other one opens a pop-up (visible in figure 4.18) when clicked. This pop-up allows to edit the box : its type, content, hint ("indications pour l'étudiant en cas de mauvaise réponse") and if it's anchored ("ancrage de la boite") or not. The hint, as we said before, is not functional since Oscar doesn't have a training mode in the version we worked on. About the anchor, if a box is anchored, it means that it will not be able to be moved by the student. We can imagine a teacher who wants to put a pre-assigned box at some place in order to give an example. He can then anchor this box to the current position.

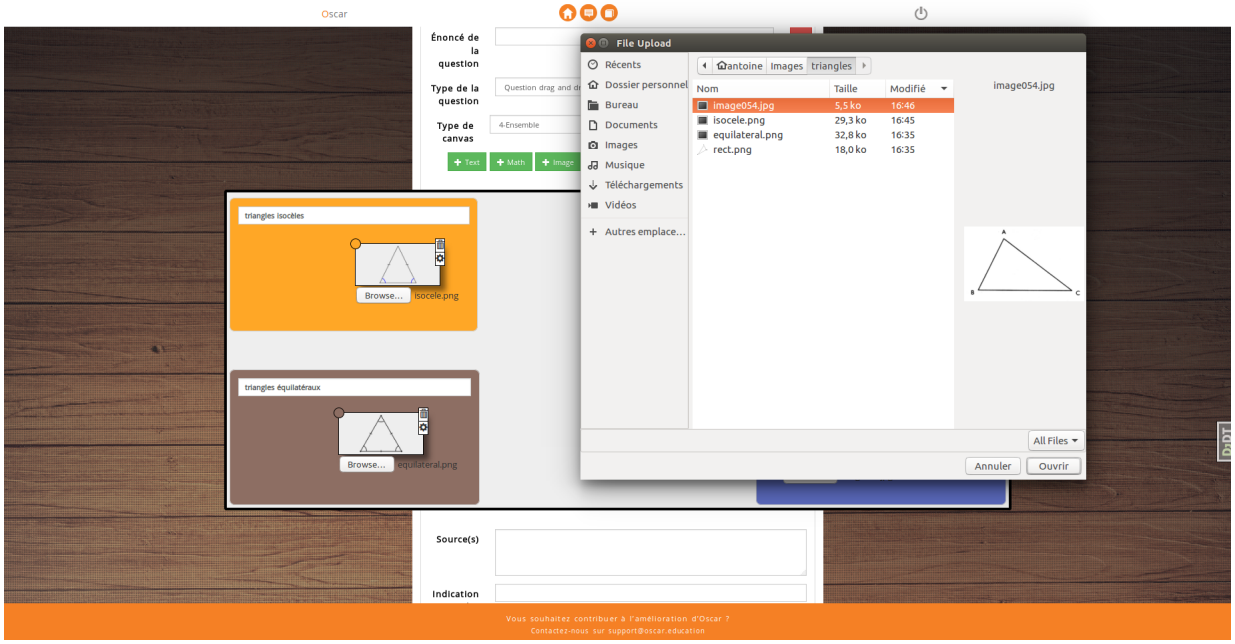



Figure 4.15: Uploading of an image

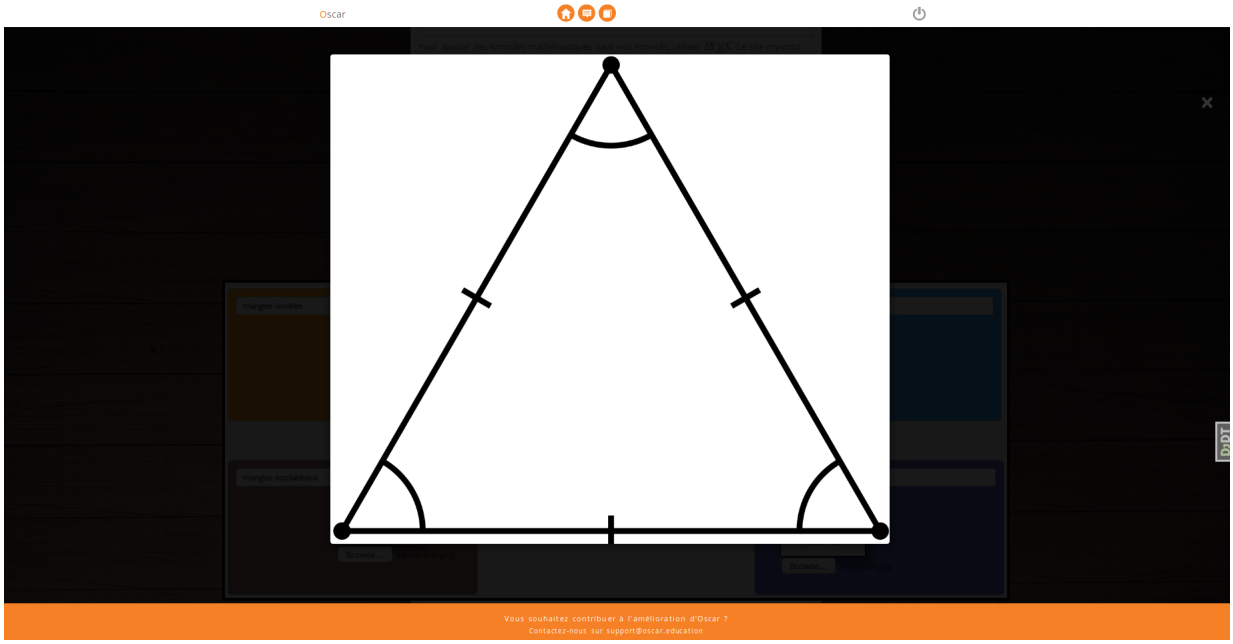



Figure 4.16: Image zoom

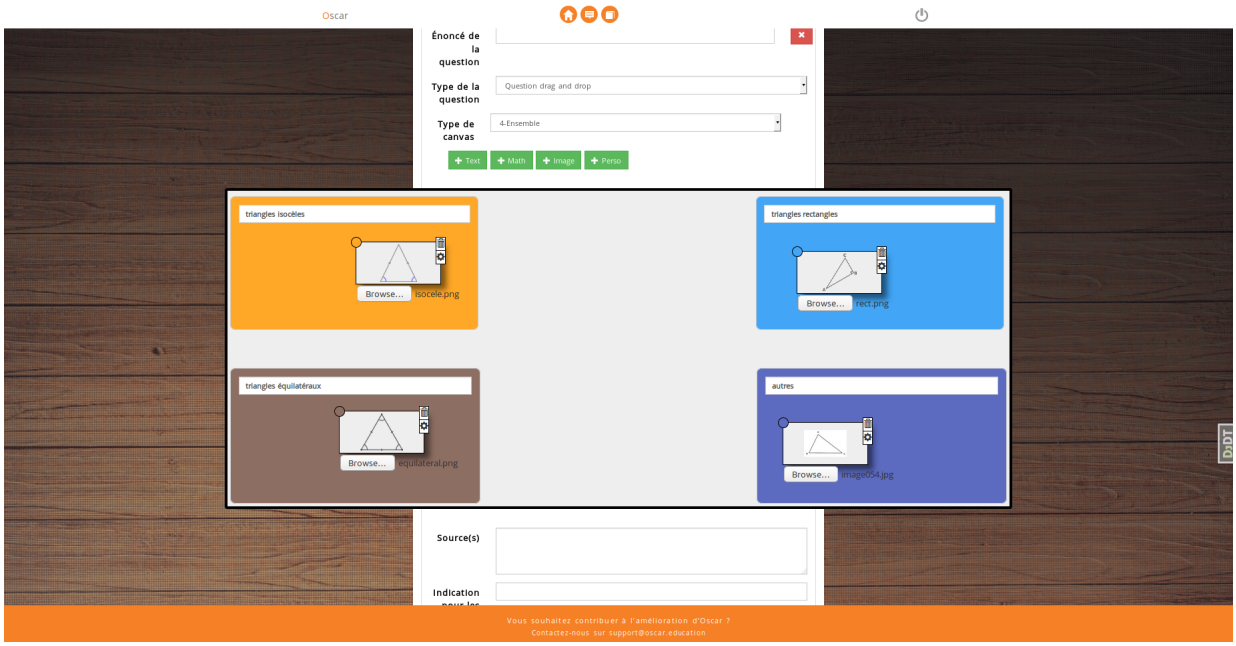



Figure 4.17: Creation of a question with images


Figure 4.18: Blank pop-up

### 4.5.3 Student's side

Once a student logs in the website, he arrives on the page shown figure 4.19. When his teacher creates a new test for his class, they are shown on the right of the page of the student and he can access it. Up to this point, this is part of the Oscar website. Our module takes control when the student begins the test.

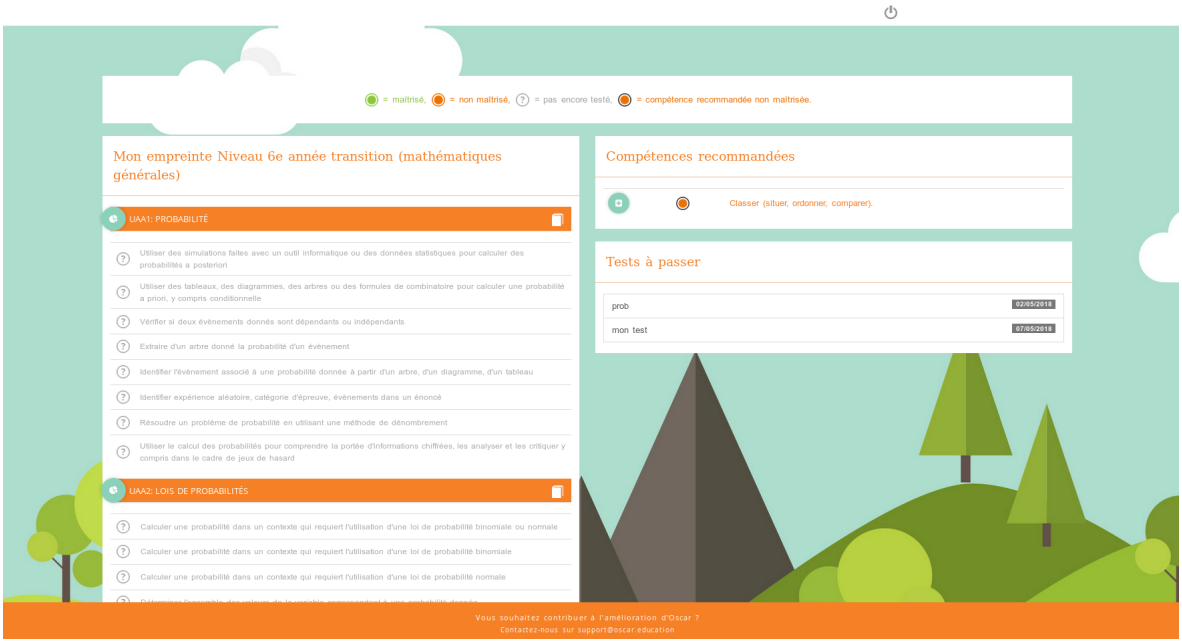



Figure 4.19: Home page of a student

An example of a set exercise in progress is shown figure 4.20. Since we designed our module in a "what you see is what you get" way it looks exactly like the interface of the teachers.

Moreover, when a student passes a test, the website will react according to the result. If the student fails, he will have recommendations for the concerned topic on his homepage. An example of such a recommendation can be seen figure 4.19 in the "Compétences recommandées" (recommended skills) section : here the student failed our exercise concerning this topic, therefore the website advise him to deepen the theory.

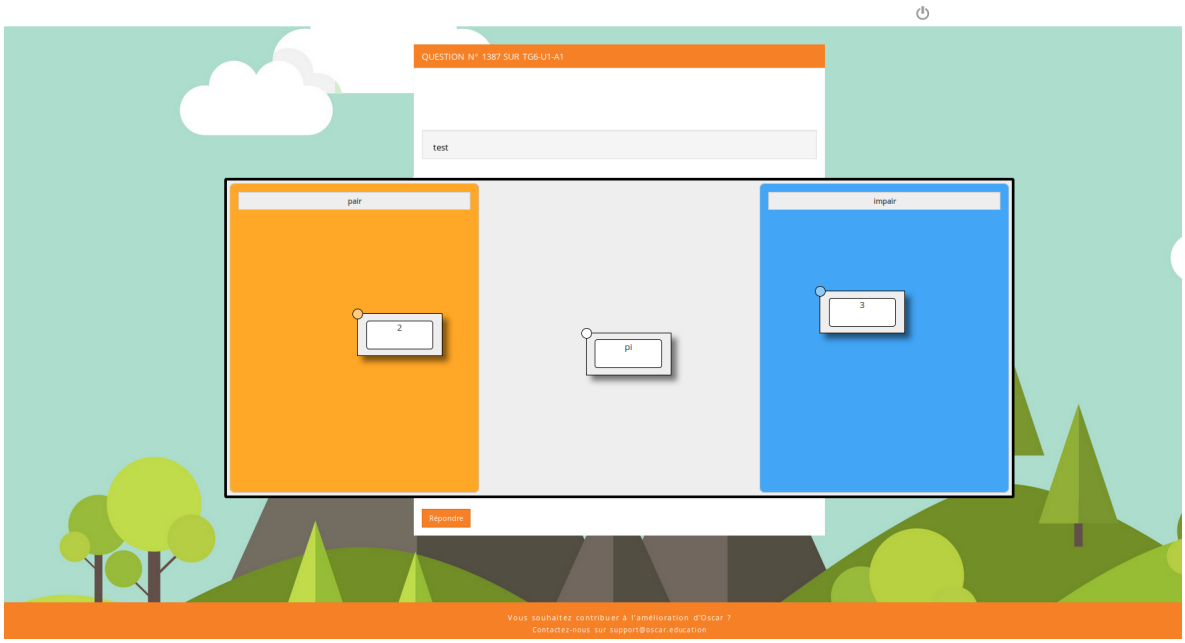



Figure 4.20: Example of a 2-set question

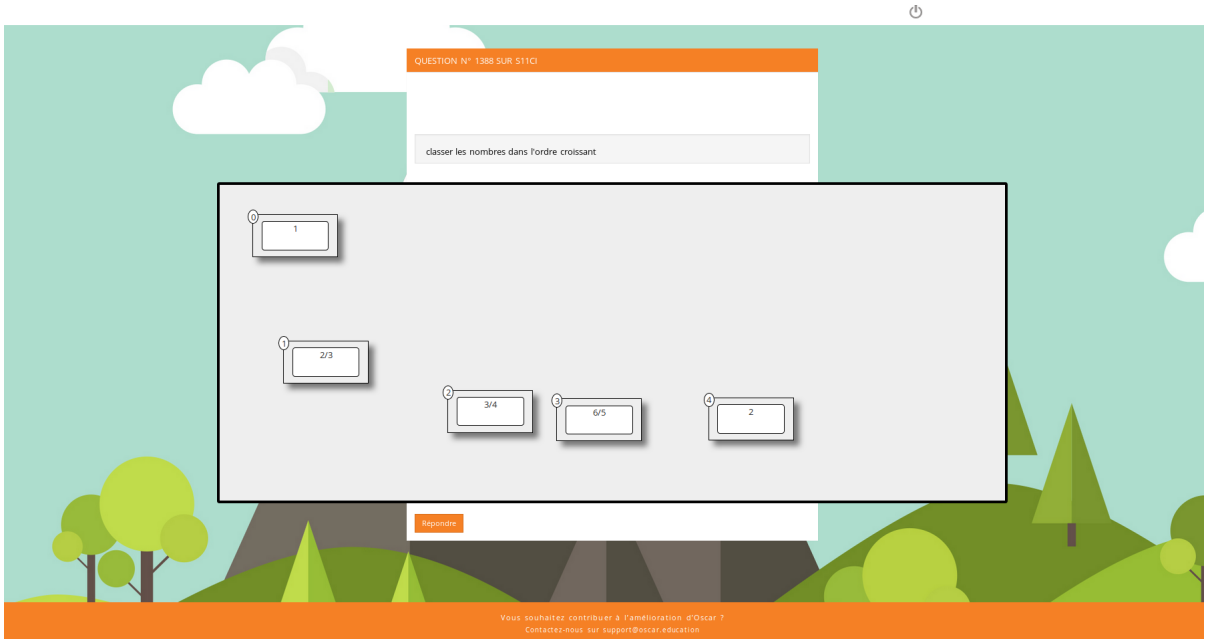



Figure 4.21: Example of an ordering question

## 4.6 Architecture diagram

To give a better understanding of the global work established into the different parts of the website, we have designed an architecture diagram for the creation of a question and another for the answering.

Each board is a technology and each box is a process. Each process is realized under a certain technology. Often, a process launches another process which is represented by an arrow pointing to the launched process. Some technologies don't involve directly a process but influence them indirectly and they are linked directly to which technology they influence.

### Teacher's point of view

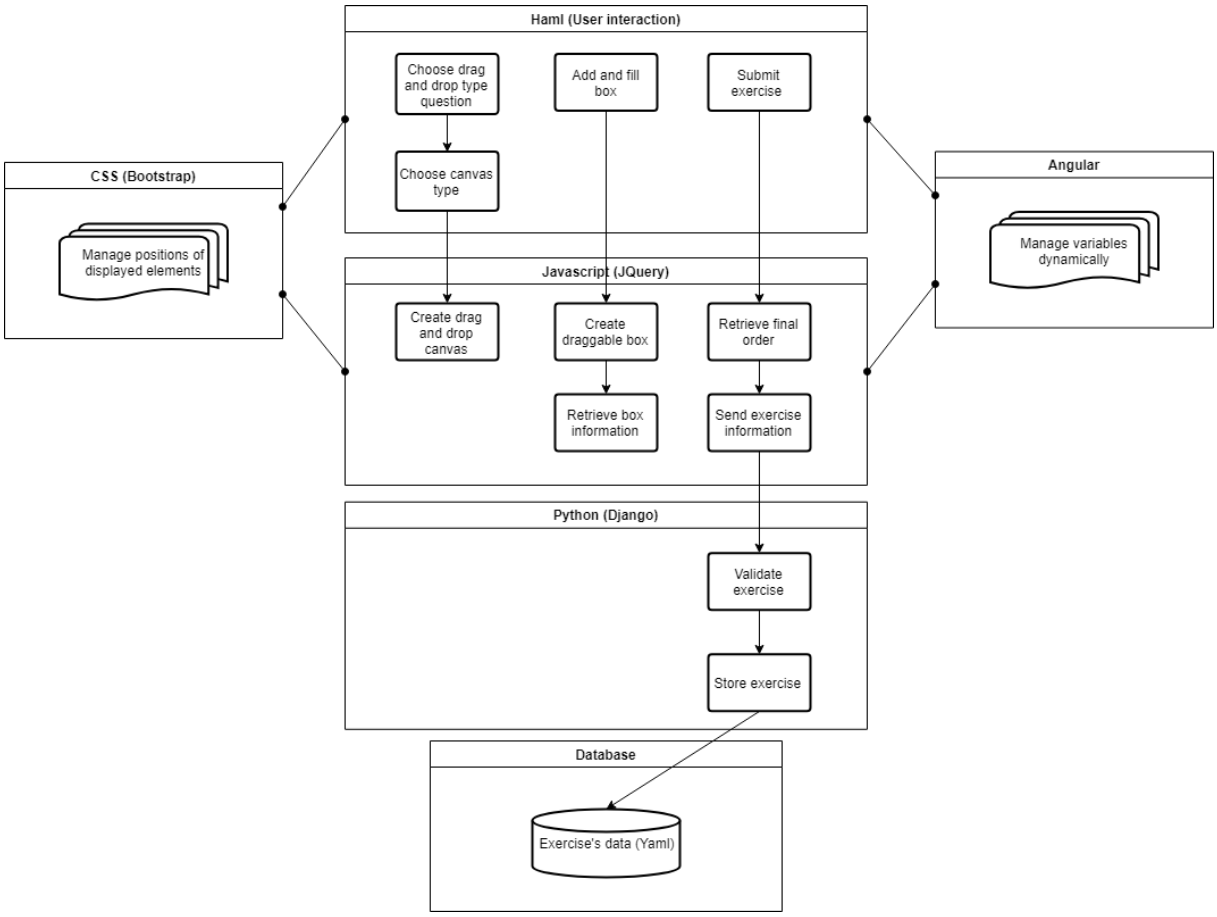



Figure 4.22: Architecture diagram of the tasks composing the question's creation

### Student's point of view

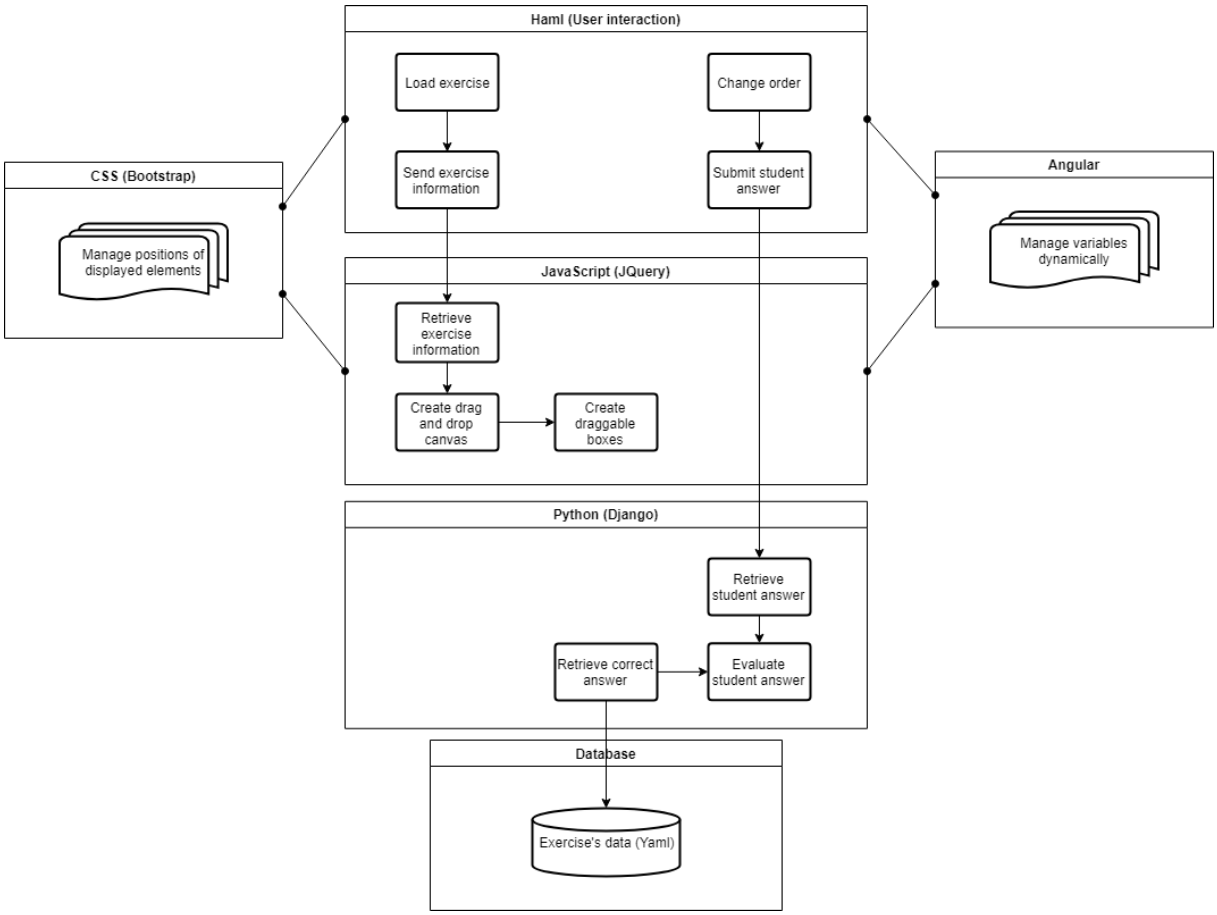



Figure 4.23: Architecture diagram of the tasks composing the question's answering

# Chapter 5

# Validation

## 5.1 User tests

When carrying out a project, one has to make sure that the result is "valid". More precisely, it is necessary to verify that what has been done matches the expectations. In our case, the most important point to verify was that the product is usable by real users. However, we couldn't trust only ourselves in this task. Indeed, the fact that we developed the product, and that we are "advanced computers users" (we are studying IT) was making our point of view biased when it comes to evaluate how easy is the use of the module.

So, for the purpose of testing the module, we've asked some "testers" to try it and give us feedback on predefined tasks. In order to have varied and relevant points of view, we tried to have "testers" with various profiles. We asked to high school student since high school students are targeted users. We also asked our promoters, beside the fact that they had to see the module for the thesis, Kim Mens and Laurent Fourny are in the education sector, thus they are also potential real-users. Finally, we asked fellow university students so we could have more neutrals points of view.

### 5.1.1 Protocol

The user-tests were logically divided into two parts :

1. Beforehand, we prepared one question for each exercise with the answers and with either texts or images as content for the boxes. The testers was told each question one by one and had to create them using our module and the different canvas types. Also, we only had them tested exclusively the parts of our module. At the moments where it was about Oscar's functionalities (connection, creation of a test...), we did the manipulations ourselves.

2. After, we gave them access to one student account previously created to answer the questions they just created. So they could feel the result of the previous work and see if it matched their expectations.

### 5.1.2 Feedback

The feedbacks we had was very disparate. Some were positive, the users generally appreciated the intuitiveness of the module along with the "what you see is what you get" part for the teachers which they found very handy. Beside that, they appreciated the interactivity, the simplicity and the overall design of the interface. Though, it might be possible that the feedback was influenced by the fact that we are students. Maybe they wouldn't have been that indulgent otherwise. Nonetheless, they were probably not totally untrue since the different feedbacks were similar in general.

Besides the positive aspects, negative but constructive criticism came along :

1. First, rather than clicking on a button to add a box to the canvas, our testers would have preferred to be able to drag them which would have been much more intuitive. However, due to a lack of time, we weren't able to implement it.

2. Then, in the 4-set exercises, a user noticed a weird behavior, when typing the title of the upper-left, he would have expected that pressing the tab key would lead him to the upper-right set title, then to the bottom left and finally to the bottom-right. Yet, from the upper-left, he was led to the bottom-right title, then to the bottom-left. This was due to the fact that we appended the set to the page in the wrong order. So the HTML made the order of the set from that and therefore, the order was upper-right, bottom-right, bottom-left and finally upper-left. This minor problem is the kind of bug that you hardly detect and that can be more easily detected with these kind of tests.

3. Next, a more annoying point raised by all the testers and that we were aware of was that when you dragged an image box by clicking on the picture, the zoom would automatically be triggered even though they were only trying to move the box. We left it like that because we thought this was something the users would deal with it, but we were obviously wrong. We fixed it by deactivating the possibility of dragging on the image.

4. Also, the testers raised some possible enhancements such as little indications for the user in order to make him understand the purpose of some elements or how it works.

5. Finally, some testers noticed minor bugs, for example, one which occurred with the pop-up. When changing a text-box to an image box, the upload of the image became impossible.

# Chapter 6

# Step back

In this section, we're gonna step back and analyze our whole project and in which way we can improve ourselves in the future.

This thesis taught us a lot. The whole process of creating and integrating a module to an already existing website is very challenging. Of course, with the benefit of hindsight, we realized what we have done wrong, what we have done right, what we should have done, what we shouldn't, ...

## 6.1 Set backs

Our first big mistake was to begin the implementation without having the code of Oscar. As we said before, it increased the workload and thus made us loose a lot of pure development time. Another consequence of this, which is also due to miscommunication with the client is that since we already worked with Oscar before, we assumed that the technologies used by the website didn't changed by then. Yet, Oscar being in development, they did change the architecture of the website and its technologies.

A way to avoid those kind of problems would have been to communicate a lot more with the client and his team. Even though it's not as simple as that since both the Oscar team and us had very tight schedule, we clearly could have improved the communication and made it more efficient.

## 6.2 JQuery

We used the JQuery library for our drag and drop behaviour. While its combination of simplicity and freedom of action was perfect at the beginning, its first characteristic became a bit blocking at some point when we wanted more sophisticated results.

With hindsight, we can say that a more sophisticated framework could have been better for our needs. Of course it would have probably take a longer time to get used to it, but we probably would have had an easier time to implement sophisticated tasks.

Furthermore, we learned during the project from the Oscar team that it was one the reasons for which they were switching to the React framework

# Chapter 7

# Further development

Even though we are proud of our work, it can obviously be improved. In this section we will talk about what we wanted to but couldn't, either by lack of time or by lack of skill.

## 7.1 Functionalities

As we said before, we spent a lot of time on the resolution of problems and thus less on the creation of content. One of the things we regret the most is the lack of content and variety in the exercises. Indeed, in its actual state, only three types of exercises are available. With more time, we could have developed more.

For instance, we thought of an exercise of "matching pairs" where the student would have to link blocks two-by-two, for example with on one side shapes and on the other their descriptions.

Another idea raised by Laurent Fourny was to have the possibility for the graduated line to have precise points on the line rather than intervals.

## 7.2 React

In order to have our module working on the current release of Oscar, it should be adapted to the React library used in the new version of the website.

The used library in our implementation, JQuery, is a JavaScript library for the manipulation of the Document Object Model (accessing and modifying existing HTML elements). While React is a library for designing and rendering user interfaces, it works through a "virtual" Document Object Model.

As JQuery is fairly different than React, it's hard for us to pinpoint what exactly need to be changed. Every JQuery function should be replaced by its equivalent in React and modify the functions when it cannot be directly changed into React language.

## 7.3 Integration

Although our module is mostly integrated in Oscar, some enhancements could be made in order to make it fully integrated. For instance, when creating an exercise with Oscar, one of the features is that the teacher can preview his exercise. This is not possible with ours. However, since we developed the module in a "what you see is what you get" way, it wouldn't make much sense. But if you have other types of exercises, it might be convenient to be able to see the whole thing. Another feature that we didn't implemented is the editing of an exercise. Indeed, Oscar allows teachers to edit their exercises as long as they don't put it online.

## 7.4 Graphical Interface

Even though we tried to both follow Oscar's guidelines and having a user friendly interface, our module could have an improved design. First, the drag and drop behavior is pretty basic, that's one of the flaws that we encountered when using JQuery as we said before. Furthermore, our use of space is not optimal, we could have a bigger canvas, a better use of the spaces of the boxes, ... Also, we could have better animations, fluidity, boxes with inertia and other things like that. One of the possible solutions would be to switch to a more sophisticated framework such as React which would kill two birds with one stone since it could help the integration into to the final version of Oscar

## 7.5 Flexibility

When developing a creative tool, there is always a limitation in terms of flexibility. On one hand, the more the creator has a freedom in what he makes, the better his creation might be. on the other hand, the more flexible the tool is, the more complicated it is to develop every part and to ensure that everything works perfectly. In our case, one of the ideas we had at the beginning was to make a very flexible tool with for example the possibility for the teacher to create his own set, his own canvas, where he could define the correction process. However, the limitations we had (the fact that we were two, with limited time, ...) prevented us from making such a feature.

# Conclusion

This thesis was rewarding in many ways.

First, one of the main lessons we learned came from the fact that we did a whole project from A to Z over a quite long period of time. This forced us to have an optimal organization, indeed, we had to divide the work in precise steps (requirement analysis, development, integration, ...), take many variables into account (the will of the client, our lack of knowledge in the used technologies, ...) and spend our time adequately since in the same time we followed other courses and made other projects.

Secondly, one of the main characteristics of this project was that we had to graft ourselves on an already existing system. This is, in general, very challenging because having to take the work and the logic of others and integrate yours often leads to various problems that you have to fix with compromises in order to have a working result. This is a very important lesson that we learned from this work.

Thirdly, what strongly motivated us in this project was the fact that we worked on a "real" project that has a real use in real life. Moreover, as student, we were really thrilled by the fact that we worked in the education domain, this made us even more glad to contribute (even at our scale) to what we think can make real difference in this field.

In conclusion, we learned a lot through this project, we worked for something that "means something" to us in the education domain and we obtained a result we are proud of. We also hope that in the future, our work will serve us, the Oscar team and potential future master's student who could continue our work or at least take it as a base.

# Bibliography


[1] Oscar website, http://oscar.education/site/, 20/05/18.

[2] *New Jersey Mathematics Curriculum Framework*, New Jersey State Board of Education, 1996.

[3] *Measuring what counts, a conceptual guide for mathematics assessment*, Mathematical Sciences Education Board National Research Council, Washington (DC): National Academies Press (US), 1993.

[4] *Understanding the role of reasoning ability in mathematical achievement*, Caren A. Frosch and Victoria Simms.

[5] *Assessing 21st Century Skills : Summary of a Workshop*, National Research Council (US) Committee on the Assessment of 21st Century Skills, Washington (DC): National Academies Press (US), 2011.

[6] *L'aprrentissage du raisonnement*, Emery C. supervised by Mister Barraud, 2003.

[7] *Mathematical competence assessment of a large groups of students in a distance education system*, Genoveva L. and Eduardo R.

[8] Learning Apps, https://learningapps.org, 20/05/18.

[9] Adobe Captivate example, https://www.linkedin.com/learning/captivate-9-drag-and-drop-interactions, 20/05/18.

[10] Educational website "Tolearnenglish", https://www.tolearnenglish.com, 20/05/18.

[11] Duolingo Apps, https://www.duolingo.com/, 20/05/18.

[12] What is Bootstrap, https://getbootstrap.com/, 20/05/18.

[13] Definition of AGILE methodology, https://en.wikipedia.org/wiki/Agile_software_development, 20/05/18.

[14] Definition of web framework, https://en.wikipedia.org/wiki/Web_framework, 20/05/18.

[15] Definition of Django, https://en.wikipedia.org/wiki/Django_(web_framework), 20/05/18.

[16] The functioning of Django, https://www.researchgate.net/figure/Main-components-of-the-full-stack-with-Django-framework-being-at-the-core_fig2_279198179, 20/05/18.

[17] Definition of Model-view-template, https://www.edureka.co/blog/django-tutorial/, 20/05/18.

[18] Definition of Model-view-controller, https://en.wikipedia.org/wiki/Model–view–controller, 20/05/18.

[19] What are the different parts of the Model-view-controller, https://djangobook.com/model-view-controller-design-pattern/, 20/05/18.

[20] Definition of HTML, https://www.w3schools.com/html/html_intro.asp, 20/05/18.

[21] HTML/HAML, http://haml.info/tutorial.html, 20/05/18.

[22] Definition of CSS, https://en.wikipedia.org/wiki/Cascading_Style_Sheets, 20/05/18.

[23] Definition of JavaScript, https://www.makeuseof.com/tag/what-is-javascript/, 20/05/18.

[24] Example of a Bootstrap form, https://getbootstrap.com/docs/4.0/components/forms/, 20/05/18.

[25] Drag & Drop library within JQuery, http://jqueryui.com/draggable/, 20/05/18.

[26] React framework, https://en.wikipedia.org/wiki/React_(JavaScript_library), 20/05/2018.

[27] Angular, https://fr.wikipedia.org/wiki/AngularJS, 20/05/2018.

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve **www.uclouvain.be/epl**