SmartBoards

André Baltazar
Instituto Superior Técnico
andre.mauricio.baltazar@tecnico.ulisboa.pt

ABSTRACT
Student motivation is very important to excel in school. However, it is hard sometimes to have this motivation. In some contexts, a technique called Gamification, is being increasingly used to try to increase motivation. This uses motivational power of games, to motivate the users by applying game elements to non-game contexts. This is used by the course of Multimedia Content Production (PCM) at Instituto Superior Técnico (IST) to try to engage the students. Although a study suggests that the students perceive the course as more motivating than others, other studies suggest that applying gamification needs to be approached carefully, because of possible negative effects. To try to minimize these effects on certain people, we developed a web application capable of presenting different content to different people. It is modular and configurable, and allows to change the presentation of each page easily. To evaluate the usability of this functionality user tests were conducted. The result shows that the users liked the way the system worked, and in the System Usability Scale test the score of the system was positive with a mean of 81.

Author Keywords
Gamification; education; motivation; view differentiation; web application.

INTRODUCTION
Motivation is a key factor for the engagement of the students at school. However, it is sometimes hard for them to get this motivation and excel in their courses. This could be caused by a number of factors, one of them being that the school system has not changed much in decades. Some people have been experimenting with different approaches to teach students. One of these approaches is the usage of game elements in non-game contexts to provide an engagement for the users. This technique is called gamification and relies on the motivational power of games, more specifically the different game elements in these games, to motive the users of the gamified context. Examples of its application can be found in many different fields, from workplaces to educational environments. The Khan Academy (https://www.khanacademy.org/) website is one of these educational environments that relies gamification to help its students learn. It uses points and badges, two common game elements present in other gamification contexts, to reward the student for learning. This kind of reward creates extra motivation for the students what already wanted to learn about the subject and further engages them to continue learning.

At Instituto Superior Técnico (IST), the course of Produção de Conteúdos Multimédia (PCM) uses gamification as a part of the course to try to engage the students. It is perceived by the students as more motivating that the other courses, however the platform which was developed and improved over the years to support this gamification had a few problems. Adding and maintaining the different aspects in this system was incredible difficult because everything relies on a single python script which generates the pages viewed by the user. There is also a lack of a back-end, for the professors, to monitor and configure all the aspects of the course.

The objective of this work was to create a new solution to replace this old system, with a more interactive site which allows to configure everything that is displayed to the users easily and allowing to present different content to different users. Having this differentiation allows to display the best information for each student in order to keep them motivated, for example, displaying the leaderboard in a different way in order to decrease the perceived competition that it may cause.

In order to know more about these effects and possible solutions that the system may need to support we had to investigate related work, which also provided a more in depth look about the various aspects of gamification, for example, its negative effects.

RELATED WORK
It is important to define what exactly gamification is. According to Deterding et al. [1] the first documented use of the term dates back to 2008 and they define it as “use of game design elements in non-game contexts”. They reckon that if games are designed for the purpose of entertainment then using game elements in non-game contexts should make them more enjoyable and engaging as well. There are a lot of examples of the use of gamification. One of the most widely used example of gamification is Foursquare, which rewards users with badges for visiting specific real world locations. Other services, use gamification to promote cooperation between users, as is the case of the Stack Overflow, that rewards users that help other users.

Gamification has been increasingly used in many different contexts, however there are some questions about its potentials and limitations. In an article with three pieces by different authors who have studied gamification, Deterding[2] challenged these authors to answer about those questions. In the first piece, by Judd Antin, he said...
that “gamification has a long way to go to achieve its potential”. In the second piece, Elizabeth Lawley expressed that implementing a gamification system can fail to engage players and possibly damage the existing interest and engagement with the service. Finally, in the last piece, Rajat Paharia talks about intrinsic motivation, and the necessity of an existing intrinsic value in the activity being gamified, and that gamification should deepen the engagement and desire to participate. Deterding concludes that gamification should be taken seriously and the needs of the users should be taken into account, the deployment of a gamified system should not just be implementing common game features and gamification should be a “means to identify and facilitate the motivation behind desired activities, using game design as one guiding lens among many”.

When designing a gamified system, we should make sure it will not harm the user in any way. Knaving and Björk[3] suggest how to approach gamification and avoid common issues. Applying gamification to a non-game context generally requires the designer to encourage some aspect of the activity in order to simplify the experience, but this simplification can encourage suboptimal behavior or obscure the activity, steering the user from the main activity because they are too focused on the gamification. Having a gamification layer can harm the intrinsic motivation of the activity if the gamification acts as an extrinsic motivation for the activity, in the case the user is not willing to engage an activity with few or no advantages in real life. They suggest that the gamification layer should be opt-in or invisible, should not spam unwilling users with information, and needs to be motivating and make the user feel competent and autonomous allowing the possibility of exploration allowing “to find and develop intrinsic motivations related to the activity”.

To try to tackle the gamification design problem, Meder and Jain[4], suggested the creation of a generic model based on the user interaction that could help assign a game design element to each user. They do this, because they reckon that since the players respond to game elements in a different way, the use of the same game elements to cover all players could have a negative influence. They suggest that players could be segmented into groups with players with the same wants and needs, and to each of these groups, an appropriate design element could be assigned.

**Education**

In education there are some example as well, like the Khan Academy, mentioned before, or the Codecademy, which also uses points and badges to track the students’ progress. More specifically in universities, there are some courses from universities around the world that employ gamification to motivate the students to have better grades, or complete optional exercises, for example. This is the case discussed by Domíınguez et al. [5] where they designed a e-learning platform and collected data on a university course using platform. Their main objective was to “increase student motivation towards completing optional exercises thought the use of rewards and competition mechanisms”. The gamification experience in this platform consisted in the presentation of a leaderboard that ranked the students according to the tasks performance. However, the students did not find it fun to compete against their classmates and suggested the introduction of more cooperation and social mechanisms to address these issues. In the end, they found that the students who participated in the gamified experience, got higher scores at practical assignments while performing worse in the written assignments.

There are other professors who used gamification applied to their courses. At Indiana University, USA, professor Lee Sheldon [6] used gamification in a Game Design course, and said that the “students performed a full letter grade better in the course than students had under the traditional approach”. In the University of Michigan, Professor Cliff Lampe [7] also employed gamification to engage his students. Instead of just applying game elements like points and badges, he uses Live Action Role Play and Guild Quests to engage the students, which allow more freedom, and even have the option to “skip tests and focus on more artistic based assignments”.

At IST, as discussed before, the course of PCM makes use of gamification as well. There are a few studies relating to the gamification of the course, where Barata et al. [8,9] discuss the results and other aspects of this gamification. They collected data for five years, with the most recent two being in a gamified context. These data contained attendance to lectures, posts, downloads of support material and grades of the students. The results show that the gamification did not have an impact on attendance but had a significant impact in mean posts per student. The final grades were significantly different for the second year of gamification where the mean and minimum grade increased. Using the data, the students were classified into different categories based on their experience points. This allowed to identify three different types of students. These types separate students with different amounts of experience points and possibly different motivation towards the course, which for the context of this project, allows us to define the requirement of needing the differentiation of views, to fit the needs of each of these types of student.

**Game elements**

To know a little more about the different game elements applied to the course of PCM, we focused mainly on Leaderboards, Points and Badges, which are used by the course. These game elements can be done in many different ways, for example, the leaderboard could show everyone, or just some part of the users enrolled in the system. Butler [10] conducted a study to understand the effect of leaderboard ranking on the perception of fun. The users were randomly passed in one of three test conditions. The first condition, they were on the top of the leaderboard, the
second they were at the bottom, and in the third they were alone in the leaderboard, with no other user ranked. The results of the study show that the users ranked on the top had a polarized perception of fun, with some users having fun and others not so much. To the contrary, being ranked at the bottom, centralized the perception of fun.

To understand how and why badges are valuable and useful in a social media context, Antin and Churchill [11], conducted a study where they present the “five [primary] social psychological functions for badges social media context: goal setting, instruction, reputation, status/affirmation and group identification. The authors suggest that the “goal setting [function] is most effective when users can see their progress towards the goal”. The “instruction” function is identified because badges show “what types of activity are possible within a given system” which exemplifies “the types of activities and interactions that are highly valued” in the system. The badges also show the interest and engagement level of the users, this is the reputation function, which linked with the status function advertise the user’s accomplishments and can be motivating. Having the ability to show the interests of the user and the activities that were required to earn the badge leads to the “group identification” because the users can see who else has shared that experience. However, they state that work should be done to examine if badging systems are engaging and motivating for all, because evidence suggests that they are not universally appreciated or understood.

**Discussion**
Overall the studies agree that gamification has the potential to motivate people to achieve something. However, there are some problems as referred in the article by Deterding, which suggested that some users fail to engage with gamification, and the gamification context should already have intrinsic value for the users to engage with it. Knaving and Björk reinforce that idea by saying that the gamification should not be presented as an extrinsic motivation, in the case it has little or no reward in real life, which could harm the intrinsic motivation. To tackle this problem, we can look at the study by Meder and Jain, which suggested the creation of a generic model based on the user interaction to split the different users into groups, which see different game elements.

This is our objective, creating the system to support this differentiation. The focus is on first supporting the different visualization of the different elements, but implementing the system in a way that in the future it is easy to integrate with a system that can assign the different views automatically based on user metrics. Allowing this extension, also opens the door to the creation and modification of the different game elements easily. With this it is possible to adjust everything, so that the players have the best experience they can.

**SMARTBOARDS**
The SmartBoards system is a web application developed with the intent of being modular, configurable and allowing the presentation of the different pages of the system in different ways to each user. This enables the professors lecturing the course, to configure the different pages to fit the students' needs. An early version of the SmartBoards was used as a replacement of the presentation part of old system used by PCM, which brought some new features to the users.

This old system, consisted of a Python Script, what processed all the information of the course, and generated static html pages which could be accessed, like the Leaderboard (shown in figure 1). To understand how the new system should work, two prototypes were created, a non-functional, which were images that show the different pages planned for the system, and a functional prototype which showed the possible interactions with the system.

![Profile Page View](image)

Figure 1. Profile Page View.

The functional prototype was developed very quickly in order to understand what was needed to create a system with the requirements we needed. The implementation allowed to see the interactions in the leaderboard and personal profile pages, and what would be required in terms of technology (ex: AngularJS). To create the prototype, the standard web technologies were used, HTML, JavaScript and CSS, with addition of a few libraries and frameworks that helped in the fast development. Twitter's Bootstrap was used to create part of page's style while AngularJS and JQuery were used to create the interaction needed for the application. The data used as a placeholder was generated using a script which served as a back-end of the system.

After completing the functional prototype, we stared implementing the first version of the system to be deployed as the replacement of PCM in the second semester of the school year of 2015/2016. This system was rapidly developed in order to meet this deadline with the same features of the old system, and some new ones. One of these new features was notifications for completion of something in the gamified experience of the course.
Architecture and Technology

In order to make the system modular it was divided into three main components: front-end (the web application), back-end (the server application) and data source (the student information). The first two parts are intersected by the modules which can perform modifications an implement features very easily.

The front-end is the presentation of a page to the user, everything they see. This is possibly the most important part of the system because it is bound to the objective of implementing the system, the differentiation of the pages for each user. Its implementation was similar to the functional prototype, however, it did not use Bootstrap for styling and some modules used to speedup the development of the prototype were abandoned because they would make the implementation of the system much more complicated.

To serve the pages of the system, we have the back-end. This part of the system takes care of processing all the requests done by the web application. It uses the data provided by data source to fulfill the requests and processes information to the login, for example. To make this system run on the IST’s Sigma Cluster, which is where the web pages of everything related to IST are hosted, we needed to make the system using PHP, which runs on the servers. To store the data in the back-end we started by using a small key-value flat file database, because it allows for an easy access and modifiability.

All the information relative to the course, comes pre-processed from the old system, which is what we call the data source for the system. These data come from different sources from Google Spreadsheets and flat files to a custom request to a page that processes the participation of the users in the course. The data is then processed according to the rules of “the game” of PCM, and outputted in its processed form which the SmartBoards system can understand, effectively bridging the gap between the data and what needs to be shown. This makes it easier to focus on the presentation part of the system without having to tackle all the problems of processing the course's rules.

The System

As referred before, the system was rapidly developed to have a minimal working version as soon as possible, to be used be the course of PCM. Since it was developed so fast, it had a few problems from the start. As was expected, in the first days of the usage of the system in the course, the system was broken a few times, but these bugs were quickly corrected to make the system work. However, there were a few problems what were harder to fix, based on design decisions made at the start of the project.

Database

One of these decisions was made in relation with the database. In order to make the system flexible to new game elements, has discussed in the related work, it was decided to use a key-value database, which allowed more freedom to create keys easily without having to use complex functionalities of relational databases, like alter tables. In reality, there are different databases for each course, and other parts, because of a problem, which is described below. The courses available and general information of the system is stored in a configuration database, which has all the basic information of the system. To store the users, we have another database which has information like name, email and username of the user. For each course, there is also a database, which has the information related with the course.

To access all this information, a Data Schema is registered in the system by the different modules. This allows the system to know what fields are available, and with some helper functions, it is possible to use the database as a whole, without having to know in which file the information is located. Defining this schema, also enables to control what other people see as public information from a certain module, because the module could have more data stored, and choose to register the schema for a particular part of the data providing some security for the other part.

The problem with these databases, was that they were implemented using flat files, to be able to run in IST’s Sigma, which does not support any key-value management system. At first, there was no separation as described above, and there was only one file that contained all the information. That file grew in size over the first weeks of the course, and with so many update requests coming from the different users, the update of the file could not be done quickly enough. In order to solve this problem, we thought about moving the whole database to a relational system, however this had a few problems. First, it would take a lot of time to migrate all the data and create the appropriate tables to hold the information. Since the system was in active development, the information could change, and we would need to make changes to the database as well. The second problem would be the loss in flexibility of using the key value databases, which allow to register the fields we want without having to bother to create a table, etc. This would also require us to change the internals of the system to use the relational approach instead of using the objects we were using to manage. There is a solution to these problems, it is the use of object mappers, which allow to map tables into objects for the system, however, with some research and failed experiment of trying to create one of these mappers, these were ruled out and we moved on.

In other attempt to fix the system, we choose to split the big file into different files, to make it easier to update, as it would not need to update every file, every time a request was done. This managed to solve the problem of the failure of the system because of corruption of the file, but created another problem, the system became too slow because of the number of files it had to access. IST’s Sigma is slow when it comes to accessing files, which made the SmartBoards pages take a lot of time to load. To finally try
to solve this, we migrated these files to a relational database, with one table with two columns, one for the name of the file, and one for the content of the file. This may seem a strange idea, but actually offers some speedups, even though it still takes a little time to load the pages.

A real solution should be created in the future. This could be actually migrating the data to a relational database using an object mapper, because it may offer a better speed at the cost of some development time implementing a proper way to allow the same extensibility that the system currently has, to define arbitrary fields easily. An easier way could also be considered, by asking the Sigma technicians to install a caching system for PHP, which would allow to cache the information for easier access. This would certainly improve the speed of access to the information and therefore the speed of loading of pages.

Settings
To allow to configure the system, there are two different settings pages in the system. The first one, is the general settings. This page allows to manage the courses (creation, status and deletion) and assign administrators to the system. It also offers a solution to invite new people to be able to access the system, allows to control the Themes (described in the next section) and allows to manage the External API settings.

The other settings page, is the course settings, which exists for each course, and is accessible to the professors. In this page, they are able to configure everything related to the course. It allows to define the landing pages for the users, configure user roles, manage the modules and their configurations, and also provides a page with information about the External API specific for the course.

Themes
The themes allow for an easy configuration of the system looks like. They are defined to allow to change the look of the system. For example, during the Easter, a theme could be used to change the appearance of the system, to include Easter related images. This is a small feature, which allows to have a different look for the pages, in different occasions, which could make the system more fun.

Roles
In order to differentiate the users, we needed some type of system to group the users together to be more manageable. This feature is the roles system, which allows to do just that, group the users into different roles. The roles are assigned to each student, which then has different settings that are configured through the settings page, like the landing page, or have hardcoded permissions, which are defined by some module in the system. For example, only the users with role ‘Teacher’ have access to the settings page. These roles can be configured hierarchically, which make allow to an easier customization of the system.

Modules
The modules are what enabled the extensibility of the system. They are one of the most important parts of the system, if not the most. They allow to define the different game elements, pages available in the system, configure settings pages, and can even run code to change things from other modules installed in the system. One important thing that modules can do, is to store information for each user, and see the information of the other modules, using the Data Schema which was previous explained. With this, it allows the modules to display that information in their pages. For example, if a module defines the game element of “Experience Points” and registers the points for each user in the data schema, other module can see those points and display them.

Views
This module is at the core of the objective of the view differentiation for each user. This “Views” module allows other modules to define pages that can be configured to change dynamically based on the roles of the users viewing the pages. This accomplished the main objective of having different forms of presentation to different students, because it allows to specify the views for the different roles, and even for different users. The module, provides a layout editor, which facilitates the creation of the views. To configure the views, the user can use different pre-created part types, which allow the configuration of these views. It is also possible to register new parts using other modules. However, to be able to configure these views with the information that they need, it was required to create something capable of accessing this information. This is the purpose of a specialized language which was created to allow this. The Expression language, allows to access fields of the data schema easily, to configure the different properties of the parts of the views.

This layout editor is a great part of the system, however, in the start it was much different. There was no editor, when the first version of the system was implemented, so there was no necessity to have a data structure to transport the edit information from the front-end to the back-end of the system, because the different views where registered in code in the modules. This created a problem when the layout editor was introduced, and the data structure used to carry the information had to be changed. With this problem solved, and the layout editor created, it was time to start implementing an inheritance mechanism for the views, which follow the hierarchy of the roles. This allows to inherit parts of other views, which makes it easier to define the different views. It introduced other problem, because the data structure was not designed to support this. However, this was a problem for the backend, because the data structure allowed flexibility enough to add a few keys with can be used to create the hierarchy. A unique identifier was added to each part of the view, in order to identify them, and allow to know which part goes where. For the hierarchy, the views where stripped into pieces and then
reassembled using the right pieces for the view in the back-end, which allowed to maintain the data structure, and changing only the process of how the views are handled.

Another part of this problem was actually the implementation of the render of the views itself. At first, it was using the modules to know what needed to be displayed in the system, this made the data structure which holds the view information very simple. However, it was then modified to accommodate the view layout editor requirements to save the edit information in the structure. The first view editor was implemented using AngularJS directives, which made it very difficult to create parts based on common parts, as the angular directives are not ready to support easily. The hack necessary to make it work, worked great for small pages, but for pages with many different elements repeating it slowed down the page too much. This was a great first approach, but the slow render of the page is a problem, which combined with the problem of the database made the wait time of loading a page unbearable. To solve this, a new renderer was created using pure JavaScript to overcome the compilation problems of the angular directives, and this solution run very fast even for big pages.

**Leaderboard**

The leaderboard is one of the modules that was implemented for the course of PCM, and serve as an example of what is possible to do with the system. It defines a view (seen in figure 2) in which the students can view their ranks in comparison with other students, however, it improved on the old system, because it is not static as it was in the other system. It uses the power of the views system, to allow the configuration of different views for different students. It is easy to define different types of leaderboards, with only a few tweaks in properties of the parts used to create it, and for more advanced leaderboards, a few functions defined by another module, for example.

The prototype had a leaderboard which was not defined as a module, because it was supposed to be implemented quickly to show the different things in a leaderboard. This leaderboard was implemented using the AngularJS module called Angular UI Grid, which helped define the different columns easily but was later abandoned, because it was not easy to make it work for the editing part of the system. To overcome this, we started using a JQuery plugin called tablesorter, to provide the capabilities of sorting and created the tables ourselves. This allows us to have control of the table, and in the future could use other JQuery plugins like dragtable, which allows to drag-and-drop columns of the table, to make it easier for the user to configure the tables.

Profile Pages

The “Profile Pages” module is another of the modules implemented for the course of PCM. It provides a view, displayed in figure 3, which can be configured with the profile of the user, using the views system, also allowing the differentiation between users. Everything defined in page uses the default parts defined by the views system, and the charts part with is defined by the Charts module.

Side View

To have information related to the user always presented on the screen, implemented a side view, which displays information related to the user viewing the system. This module is the example of the power that the modules have over the system. The module has the capability of adding itself to other pages in the system, and display the information. It even has a group of links which are generated automatically based on some configurations of it is adding itself to. This block allows the users to easily compare their stats with other students for example.

**Quests**

The quests module, implements other game element of the course of PCM, where the students have to use the skills learned in the course to find the way to unlock the next challenge, like a treasure hunt game. It is one of the game elements which aims directly to provide cooperation between all the participants, because it rewards all the
students with the same amount of experience points, and the higher the number of students participate, the higher the amount of point earned. The module was deployed to be used in the quest this year, and received great feedback from the students which suggested various ways on it could be improved. A configuration page exists for the module, which allows to define the active quest (one active quest at a time). The levels of the quest can be configured in other page, and even allows to upload resources like sounds, to be uploaded, to be inserted in the quest, without having to have access to the server to upload the files. One great feature also implemented is the validation of levels, which require a image to proceed. The user is able to upload the image, and a comparison pixel by pixel is performed to check if indeed the user was able to get the right answer.

Charts
To provide different types of charts, a charts module was created. It registers a part in the views system, which can be used by the views, when this module is enabled. It provides four different types of chart. The line chart and bar chart, are used by the profile pages to display some information related with the experience points and badges of the users in the course of PCM. The star plot is also displayed in the profile page and shows a comparison of various user related variables, with the average of the other users. Finally, there is a more configurable progress bar, which is able to take any value using the expression language, and display it in a form of a progress bar.

Awards and Overview
Finally, there are two modules, the Awards and Overview, which simply define more pages that can be used in the system to display information. The Awards page provide a view into everything achieved by the students in the course, while the Overview, provides a summary of those things, and other information related to the user, like the last access to the system.

Game Elements
The game elements of Skills and Badges used in PCM were also specified as their own modules. They provide a link for the data between the data storage (the old script), and the back-end of the system, by registering the available fields related to the game elements in the Data Schema. These are implemented in different modules, in order to make it easy to disable one of them if necessary. Apart from registering the fields in the schema, they also provide default templates for their presentation in the view, for example the Skills module, provides the Skill Tree presented in the profile page of the student. They also register custom functions in the expression language, which allow to make the display in the views more efficient.

EXPRESSION LANGUAGE
As explained before, the Expression Language allows the administrator of the course to configure the views of the system. The main purpose of the language was to have an easy way to access the fields of the system, do operations with the values, and display them in the different parts of the views.

To start defining a syntax for the language, we needed to know where it would be used. In this case, it is going used in the middle of text to be displayed by the views. This restricts the way it should work, and everything that is typed as an expression is evaluated as a value or text. There is no need for a print function to generate output of the language, because everything that is evaluated is in fact the output. So a hello world example in the expression language would simply be the expression 'Hello World'. To evaluate anything, we use curly brackets '{}' which are then compiled and evaluated when the page is created.

The language includes all the normal operators found in common languages. These operators include the addition +, subtraction -, multiplication *, division / and modulo %. It also provides logic and bitwise operators, and && (bitwise &), or || (bitwise |), negation ! (bitwise ~) and bitwise xor ^. The common logic comparison operators are also available in the language. To use them we simply do for example, '{1+1}', which would result in '2'. The language also has the capability of executing functions defined by modules, using the symbol '%' before the name of the function and then putting the arguments of the functions between parenthesis, for example, '{$hello("world")}', which would call the ‘hello’ function with a string ‘world’ as an argument.

To access the fields from the database to be displayed, we use any word that is inside the curly brackets, and does not have a symbol preceding it, as a path in the database. If we wanted to access all the users in the system, we would simply use '{users}', which is evaluated as the map that contains the users. To access a value inside the map, we can use rectangular brackets, for example, '{users[12345]}', which would result in the object of the user with id 12345. Finally, to access a field of that object, we can use '{users[12345].name}', in which the ' ' (dot), is used as a path separator for the object, this would access the name of the user with id 12345. All these fields need to be defined using the data schema discussed before, which provides a mechanism to protect the data that should not be accessible by the views. The language also has the concepts of variables, which can be used to make it easier to configure the views.

To do all this, a grammar had to be created, to generate a parser capable of parsing the language, and evaluating it for the views. All this was done with a using a PHP port of the Jison parser, available for JavaScript, which was is easy to understand, and the API used is similar to Bison which is used by the course of Compilers at IST.

Minor features
There are some minor features that do not do much, but are important for the system to work. One of these features is a script that is used to download the photos of the students,
from the Fenix system to be displayed in the SmartBoards system, where necessary. Another script is responsible for downloading the usernames of the users provided by the old system, because the old system used the id of the student as the identifier, but for login purposes it is required to have the username.

Near the end of the development, a simple External API was also implemented to allow access to the data of the SmartBoards system by outside applications. This API was implemented to allow to change any data we want in the system, but in the future, it should be updated to restrict the way it works, to have more security. Information related with the usage of the API is available with the documentation provided with the system.

**Effects on Motivation and Engagement**

The new features that allow for the differentiation of views for the different students were introduced to allow the study of the effect on students’ motivation and engagement. Using the research done by the various studies presented in the relation work section, it is possible in the future to use all the features provided by this new system to study the effects of the different types of leaderboard and functionalities that were appreciated by the users in those scenarios of gamification. Features like the implementation of notifications of badges, that can possibly have the effect of instant gratification on the users.

**CASE OF PCM AND POTENTIAL**

As discussed before, the first version of the system was deployed for the course of PCM. This allowed us to have the system running with users, which also allowed us to catch some problems like the database problem. In the following sections we discuss the specific configuration for PCM and the potential of the SmartBoards system.

**Course of PCM**

The case of PCM was the case for which we developed the modules to work. It uses the Leaderboard, Personal pages and the other game elements developed specifically for the course. It uses the Badges, Skills, Awards modules to provide the basic game elements for the gamification of PCM. The Badges game element is used in the course to give a reward to the users when they accomplish a task, from posting on the forum and submitting their assignments, to participating in the classroom. Each of these rewards have an experience points value associated with them, which gives some points towards the final grade in the course. Some of the work given to the students is provided through the game element of Skills, which specifies a task that needs to be completed for each skill, that the user can unlock to earn some more experience points.

In terms of configurations, in the leaderboard page of the old system, students could only see their position ordered by experience points, but with the new views system, the viewers have a greater control of what they can see, with the ability to sort by the other columns of the table. In terms of what is displayed in the view, it has not changed much, it still displays the rank of the student, its photo, name, experience points, level and badges, but now also includes a Sparkline with the evolution of the experience points over time. The profile page did not change that much either, only with the inclusion of a star plot to see the experience points from the different evaluation components of the course.

All this was easy to implement with the power of the views. The introduction of the system, and some development that was done during the semester, allowed the students to provide feedback for some of the features, for example, the quest, which allowed us to improve it. Other suggestions were not implemented but can be considered for the future.

**Potential of the SmartBoards**

As hinted before, the SmartBoards has great potential when it comes to extensibility. Here, we explore the potential of what the system is capable of, and the ways it can be extended.

The Views allow the administrators to extend the system in any way they want. It is possible to do some extreme examples of views that can be configured by them. One example that both could have an impact on the student and also can be used to monitor the course, is to display which students are on a path to failure. To achieve this a view similar to the leaderboard, with a list of users could be created to display the users that do not have a very good amount of experience points relating to the days left to the course. It would show the users failing the course, if they keep doing the same amount of work they have done until that point in time. Of course this is not a great formula to compute how is going to fail and how is not, but a function with some other kind of inputs could be used to greatly improve such estimation. An example of this, is the use of the different types of students which were identified in the study of related to the course of PCM, where the students were put into different categories based on their performance. If the prediction was good ahead of time, it would be possible to display it easily.

Another great thing that has potential is the Expression language. It could be extended to allow for example the definition of functions for the views, in some text area accessible from the interface, without requiring the functions to be defined in PHP by a module. The only thing necessary to do this would be a slight change in syntax to allow the definition of functions and other mechanisms of control, like if functions and loops.

This is possible because of the modules, which provide great the extensibility of the system. There are many more ideas that could have been implemented as a module, one of them is the use of parts of the Quests module, to create a tutorial system. The various levels of the Quest could serve as the pages of the tutorial, and the interface of editing the level would allow to change the tutorial easily. Another
example, is the implementation of the Hall of Fame, which is something that PCM has in the end of the course, which displays the best work done by all the students. This could be implemented as a module, and enabled only when needed.

**EVALUATION**

To evaluate the system, a summative evaluation was performed in the end, to test the capabilities of the view editor in the system. This evaluation allowed us to measure the usability of the system, by having the users responding to the System Usability Scale (SUS) test, and by measuring different parameters during the execution of the tasks.

These tasks focused only on the layout editor of the views, in order see if the users where able to do the tasks, and the difficulty that they had executing them. There were seven tasks that the students were asked to perform. They were presented as following to the users:

1. Create a container block with the header text “Last Activity” and “images/awards.svg” image Mark table headings
2. Create a table with 3 columns and a header row (“Photo”, “Name”, “Last Activity”)
3. Turn the table into a list of course users, sorted by last activity, filtered by last activity (show users with activity only)
4. Display the photo of the user in the first column
   - Photos of the users are located in the “photos” directory, indexed by username and have “png” extension.
   - Example path: photos/ist173137.png
5. Display the name of the users in the second column
6. Display the last activity of the users in the third column

7. Format the table
   - Center the “Photo” column, and apply the predefined “user-image” CSS class to the photo.
   - Center the “Name” column.
   - Style required to center the columns: ‘text-align: center’.

To limit the time that the users had to complete the tasks, expected times and maximum times where defined for each task, based on the times of the pilot test and the difficulty of the task. The expected times were as follows for each task: 45, 90, 120, 90, 60, 60, 90. The maximum times are the double of that value.

In order to investigate the result of the test, a proper analysis was done to each of the measured parameters for each task, and the questions asked to the users. For continuous variables, the mean, standard deviation, median and quartiles were calculated. The quartiles were also calculated for ordinal variables, along with the median and mode. Finally, for the nominal variables only the mode was calculated. In the cases that it made sense, some percentages were calculated, like the percentage of completion of each task.

The tests were conducted mainly on male users (19 out of 20) with average age of 22.9 years old (standard deviation of 2.92, median 23). The majority (95%) of them have knowledge of more complex applications, that require learning something about the application and are not simply applications that are used by everyone, like e-mail applications or social media.

Around half of the users tested (55%) had used a Content Management system, which could have some benefits when using the SmartBoards system. Finally, 80% of the users answered that they had used web technologies like HTML, CSS and JavaScript.

The task completion was very good for tasks 1, 5, 6 and 7, with a completion of 100%, and for task 2 with 95% completion, while for tasks 3 and 4 it was lower, with 60% and 40%, respectively. This was expected due to the difficulty of these tasks, as the task number 3 was the first task that required the users to learn the expression language, and task number 4 used some complex features of the views and expression language. The time of execution of the tasks was also very good, nearly matching the expected times that were purposed before. This can be seen in the figure 4, with the mean times and standard deviation of each task.

With the questions asked after the test, the users revealed that they liked the system, and thought that adding blocks and other parts was very easy. In the second question, the users rated the difficulty of the Expression language as a whole, with a mean rating of 4 out of 5 and a mode of 5, which means that they did not find the language difficult to understand. The third and fourth questions were related to the difficulty of using variables and fields, which had also a mean rating of 4, but with a lower mode, which reveals the even though they understood the language as a whole, the concepts of variables and fields were more difficult to understand. The last three questions were related with the
documentation and revealed that the users thought that it was helpful (this question had the mean and mode of 5, which translated into a strongly agree), understood the fields available in the schema (a 95% number of yesses), and rated the difficulty of understanding the documentation with a mean of 4.

Finally, we used the System Usability Scale test to evaluate the system. Using the percentile ranking with the value 68, which according to many studies is identified as the mean value for the score of a SUS test, and the mean value for the system, which was 81, with a standard deviation of 11, our system is ranked in the top 11.1% of the applications evaluated with this method. The mean, mode and quartiles were also calculated for these questions, revealing that some users think they need help of a technical person to operate the system. This question had a mean value of 2 (disagree with the necessity of needing help), however, in the third quartile, the response is at a 3, which suggests that 25% of the people may need some help.

In general, the users commented about issues with the system, and gave suggestions for the improvement of the system. Some bugs were also found during the tests, that need to be corrected.

CONCLUSION
The Smartboard system is now capable of creating the differentiated forms of presentation for different students, as was our objective. The views can be created in the application, which is easier for non-technical users, even though it may require some learning, especially to understand the Expression language, for the more advanced features. It should not be very difficult, because of its simple syntax/grammar. In the future, it could be changed to something more visual, to make it easier for non-technical users. Some visual language, which requires from the user a very minimal amount of knowledge about the system to be able to configure the pages.

The results of the evaluation were also great, because the users were able to complete the most of the tasks given to them, and gave a good amount of feedback and good rating in the difficulty of the tasks. The System Usability Scale test puts out system in the top 11.1% of the applications evaluated with this method, which is great.

While the main objective of the project was accomplished, the motivation behind it still needs to be tested in the future. Modifying the views, for the students not performing well or that show negative effects with the presence of Gamification, should be approached carefully because it could increase this negative effects, making the students less motivated toward the course. It is up to the professors, or some other future system, to decide what are the markers that indicate the lack of motivation (the types of students discussed in the Related Work could be used, for example) and try to correct them using different views to inspire those people, motivate them and ultimately to succeed.

REFERENCES