I Am Here!

Matilde Nascimento
INESC-ID
Técnico Lisboa
Lisbon, Portugal
matilde.nascimento@tecnico.ulisboa.pt

***Abstract*—Tracking student's attendance in classrooms has been a problem for the professors for a long time, not only because of the difficulty to guarantee that the student is indeed the room, but also the time it consumes afterwards to manually transcript and validate the data. *I Am Here!* has the goal of being an innovative automated attendance system that aims to fulfill the existing gaps in this area, with a simple, costless, portable and fraud-resistant system using the Device Fingerprinting technique.**

A critical aspect was the design of a user interface that made the process as easy as possible for inside and outside the classroom, but at the same time sufficiently complex as to preclude, as much as possible, attempts at fraud. As a case study and form of validation, the system was deployed in a course where 116 students were enrolled, in Técnico Lisboa. The system was submitted to three evaluations: a Classroom Usage Evaluation, an User Evaluation and a Device Fingerprint Evaluation.

***Index Terms*—attendance system, I Am Here!, student, classroom, professor**

I. Introduction

In the teaching field, the most commonly used way to check if a student is in a classroom is by roll call or by forcing students to manually sign an attendance sheet that is passed around in the classroom. Both of these systems are vulnerable and not scalable for large numbers of students, being too time consuming.

With increased importance given to attend classes, such as the knowledge that is acquired (or when there in impact in the final grade), motivation to cheat the attendance checking increases as well. Because of that, some students will try to cheat the system. Rather than trying to attend the class, students try to manipulate the attendance system, faking their presence, by asking a colleague to sign on their behalf, i.e., having someone else impersonating them - which can be illegal. Also, the professor does not have the time or energy to keep track of how many times a student has signed, or to count the number of students in class (that can be constantly entering/leaving the room), given the fact that some classes can go up to 50 or even 100 students. Also, as explained, the professor has to manually check all the signatures (that can be easily forged) and possibly transcribe them to a spreadsheet; this can be a very long and tedious process.

Taking too much time during and after class, being inaccurate, with high (possible) human error, not scalable and not environmentally friendly (the amount of paper that is wasted and cannot be re-used), these type of systems are neither reliable nor effective in any way.

The goal of *I Am Here!* is **to implement an Automated Attendance System (AAS) that is efficient, fraud resilient, does not require (extra) hardware and does not waste time before, during and after class**.

To accomplish this, a website (as it is cross platform) was implemented. The professor can create attendances, according to a course and shift, and can, thereafter, check who was and who was not in the classroom, the students' attendance tendencies and can also export the attendance history, so it can be used in another context. On the students' side, the solution is based on a system where students have to insert a number of consecutive codes during a short period of time, making it very hard to cheat the system. But, in case a student manages to do that, the system can detect those cases using Device Fingerprinting. With this tool, it is possible to distinguish each student's device fingerprint or, in a fraud case, disclose two (or more) fingerprints.

To test the system, usability user tests were done for both students' and professor's side, as well as a device fingerprinting evaluation. The results show that *I Am Here!* is simple to use and understand, that can also catch the most common fraud attempts.

II. Related Work

Given the scope, it is useful to research different types of Automated Attendance Systems (AAS). The research was around systems with the same mindset as *I Am Here!*.

A. Radio-Frequency Identification (RFID)

Radio-Frequency Identification (RFID) is a technology that was invented around the 1940s as a means for remotely identifying military aircraft [1][2]. RFID systems require two main components: RFID readers and tags, and an application or software package running on a computer [3]. This has been used as an student attendance checking system in several teaching facilities. It normally has a reader in the entrance of the classroom, and students need to flash their cards to check their attendance.

B. Biometrics and Face Recognition

Biometrics and Face Recognition are two types of system that extract key features of human characteristics. Biometric does the identification and verification given physical and behavioral traits, such as fingerprints, irises, retinal patterns, fingerprints, voice, etc [4]. Face Recognition analysis key point

features of the human's face - nose, mouth, edges, eyes and other characteristics [5].

In Thailand, it was implemented the combination between Google Forms and a Speech Recognition (a biometric) system [6]. Upon the beginning of the class, the professor registers the students that arrived on time by marking check boxes in Google Form; for the students that arrive late, because speech recognition can detect numbers more accurately than words, the professor reads the last three digits of the student's number to the (phone's) microphone - students can then check on the screen that their attendance was marked.

For Face Recognition techniques, in most scenarios, a camera is installed in the classroom and it viewing angle captures all the seats of the room. Typically, the systems already has a database with photos of the students who are taking the course, and the picture taken during class is compared with the one that already exists.

C. Software-only systems

There are some software publicly available whose purpose can be to do the attendance checking in classrooms. Usually, these systems are malleable, in a way that, in a students attendance tracking, the professor can manipulate whatever way wanted.

Quick Response Code, better known by QR code, is a technology that has been widely used by many industries [7]. In the field of student's attendance system, this technology, opposed to the previous ones that required extra hardware, has a cheaper development cost, and it is easier to use. Instead of using standard attendance systems, an automated application was implemented in Malaysian Institute of Information Technology where students could generate unique (and non-transferable) QR Code with their mobile phone, that in class could be scanned by the professor.

Kahoot! is a company born in March 2013, with the purpose of improving the education around the world. Through questions that can be personalized by the professor - for example, inserting pictures - the students only need the code of the game to enter and start playing. As a game-based platform, that can be created by anyone and played by who has the code of the game, its layout engages students in a healthy competitive way. In United Arab Emirates, Kahoot! was used not only to challenge and motivate physics students, but also to check their attendance (if the student participates in the Kahoot! game has to be in the classroom) [8].

D. Discussion

RFID, Biometrics and Face Recognition, although they cover most of the fraud cases (except the RFID, where a student can bring a colleague's ID and scan it), they do require extra hardware, which requires maintenance and spend (more) money. For the software-only systems, neither the QR Code or Kahoot! have a system where it is possible to analyze and manage the classes' attendances, and these system are not fraud resilient - someone can impersonate a colleague.

In conclusion, neither of these systems has the same specification as the proposed one. Yet, it is possible to improve some ideas from these systems. In the next section the solution is presented.

III. Approach

The original problem can be summarized as *How can attendance be verified in universities in an automated manner without being cheated by the students?*. For a **costless and portable** solution, a website will be developed - so it is not required extra-hardware for the university, students nor the professor. This approach is compatible with the different operative systems that currently exist for portable devices, and because it will not require students to download any (extra) software. So that **only students inside the classroom can check their attendance**, it will be required, to take the attendance, that students type a series of consecutive random codes with a limited time to type each code. This makes the job to take the attendance for a colleague harder, because students will have just a few seconds to type each code, leaving a shorter time to type and send the code for another person.

In a scenario where a student inside the class tries to type the code for a classmate, the system will prevent and flag those attempts using Device Fingerprinting. Each student's device will have a *unique* fingerprint; if two or more attendances have a very similar fingerprint, those students will be flagged.

The proposed system will have two distinct sides: the professor's and the student's. For the professor's side, it is only required that the professor (or the classroom) possesses a computer and a projector in the classroom (usually all classrooms in universities have one).When intended by the professor the attendance checking can start. The professor navigates to the website, enters the personal credentials and chooses the course and shift that they are lecturing.

Then, a *QR Code* and a link will appear for the students to scan/type - so the students can access the website to check their attendance. From this point, it is possible for the professor to start the attendance. In case the students finish the attendance checking (correctly entering the sequence of random codes) the professor can stop the attendance. The time to type each code, the number of consecutive correct codes necessary to take the attendance, the length and type of the code (Letters, Numbers or Letters + Numbers) is defined by the professor. This definition can be done until the beginning of the attendance checking, but it can also be done at the beginning of the semester, creating all classes of a course with a specific configuration (that can be adjusted anytime during the semester).

For the student's side, the beginning is similar. The students can scan the *QR Code* or type the link that is being projected on the wall. Then, it is required that students log-in using their student's credentials. After logging in, the students will need to type a series of random codes into their devices to check their attendance successfully.

IV. Understanding Code Entry

To understand how much time an user needs to type a code, and the perfect length (with capital Letters and/or Numbers), a series of tests were made. The goal of these was to understand, for each type and length of code, how much time a student takes to input the code without failing and without giving too much time - so it does not take too long and does not leave space to send/do for other person.

The tests consisted in typing five sequential (and random) codes with five different lengths (from four to eight characters) and three different type of codes (Numbers, Letters + Numbers and Letters), with 10 seconds to type each code - giving, in total, 15 combinations. The time chosen (10 seconds), given the intended scenario, was a reasonable maximum value, and it allowed us to find, within that limit, how long did students actually take to enter each type of code. The sequential five codes were chosen as a fitting length, since more would be considered too much time taken from a class, and to understand, up until five sequential codes, how users interact. This gives a maximum overall duration of 50 seconds to do the task. In Figure 1 is an example of what the user was presented with when typing the code: time left, number of the code and the code itself.

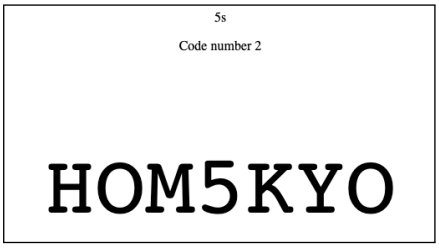

Figure 1. Example of a test; Letters + Numbers as a code type, with a 7 characters length.

Each time the user submitted a code, the system registered the code length, type of code, if the code generated was equal to the inserted one and the time left to type that specific code. After each test, two questions were asked: *Was the number of characters easy to type?* and *Was the time to type the code sufficient?*, on a 1 to 5 Likert scale, within 1 being *Strongly disagree* and 5 being *Strongly agree*.

These tests were done with 25 students from Técnico Lisboa; the participant used a portable device to type the codes, and the generated random codes were being projected on a wall - 20 of them were done using a (personal) mobile phone, and the rest with a (personal) computer.

*a) **Time**:* We conducted ANOVA tests to try and find, for each code type, if there is any statistically significant difference for the time taken by users with different code lengths (and Tukey post-hoc tests to identify where). Between the code types, there was a statistically significant difference (*F(2,1844) = 5,26, p = 0,003*). For both **Letters + Numbers** and **Numbers**, the significance was in codes with 7 and 8 characters codes (when compared with codes with less characters). Lastly, for Letters, the statistically significant difference starts from 6 to 8 characters.

*b) **Error**:* Regarding the errors, i.e., when the users did not type the correct code, 162 out of all the input codes (1875 in total) were an error; 63 out of the 162 were codes with just Letters, and 72 were Letters + Numbers, leaving the codes with just Numbers with only 27 errors. For the errors, one-way ANOVA proved that there is no statistically significant different between the three code types with different code lengths ($F(1, 326) = 0.88258, p = 0.348192$).

*c) **Questions**:* Lastly, based on the answers from the tests' questions, the results showed that (only) the second question (*The time to type the code was sufficient?*) had a statistically significant difference as determined by *one-way ANOVA* ($F(2, 360) = 7.13, p = 0.001$), and a *Tukey's post hoc* test revealed that the statistically different was in the codes with Letters + Numbers.

A. Discussion

Taking into account all the results from the tests, it is possible to conclude that the best combination is **Letters** as a code type and a length of **6 characters**, not only because the average students can type with no problem, but also does not give too much time to send or do for a colleague.

V. Implementation

In its current form, *I Am Here!* requires all users to authenticate by using FenixEdu [1] credentials. However, using a different *OAuth*-based authentication method would be trivial in a future version that wishes to implement it.

A. Architecture

The system consists of two major components: the front-end and back-end. The back-end has two components - the server and database (all the information is stored in here). The server directly interacts both with the database and the front-end. The front-end - a graphical user interface - takes the information from the server and displays it according to the visualization and interaction required for the proposed system; the information displayed differs, according to the type of user (professor or student).

B. Implementation

Throughout the testing phase, *I Am Here!* was (and still is) running in a virtual machine running in Debian [2]. The project can be checked on GitHub [3]. After researching different types of languages, Node.js was chosen due to the type of system that was going to be implemented. To have a system that is accurate and updated every second, Node.js appeared (and then proven) to be the best option, because the state of the system is stored in memory and not in the database, as opposed to other languages.

[1]The academic management system developed at Técnico Lisboa (https://fenixedu.org/dev/overview/)

[2]https://classcheck.tk

[3]https://github.com/matildepgrn/I_Am_Here

Then, for the front-end, given that the system was going to be implemented from scratch, JavaScript, HTML5 and CSS was the way of doing a simple and yet effective system. For the database, having already experience with MySQL, it was the software chosen to use as the database.

1) User Interface: In the professors' perspective, there is a couple of tables and selects; all these were done using `<template>`, a JavaScript mechanism. This helps because the number of necessary rows for each table/select are different, and, depending on the size of the information, it allows to have the right number of rows. To alert the user when is not typing the correct way, it is implemented a regex (a regular expression) to assure that each line is according to the format. The regexs were implemented and verified using an online website [4].

It is implemented the sorting functionality in tables from `W3.JS` resource. It is using the `HTML.sort()` function [5], that receives the table's id and the column that it wants to sort.

To give a visual interpretation of the time left to input each code, a countdown bar was made using only JavaScript and CSS. It is a `<div>` that each second updates the bar's filling, according to the total time of each code. The gradient from black to red is also dependent on the time, as the time is lower, the color gets closer to red.

In the beginning of the implementation, a problem was found in the input on the student's *side*. Users with *Firefox* as their browser had a problem typing the codes, where it would automatically auto-complete the "word". This outcome was not intended, given that the code is a set of letters (and/or numbers) picked randomly, so the probability of it being an existing word was low and the auto-complete would just slow the process of typing it (because the users had to keep deleting the word that was appearing). After searching about the problem, the solution was to set the auto-complete attribute to `nope`.

a) Responsive Design: Students can either use a computer, tablet or a mobile phone. Because each one of them has a different screen size, it was added the tag `<meta>` [6] to use the method `viewport` [7]. The first one specifies *metadata* from a HTML page, and the second one is a metric of the tag `<meta>` that helps to fit the information displayed according to the screen's size.

From the professor's perspective, they ((usually) use a computer. All the information needs a minimum size screen, but the most important part is the projector (which is where student will be able to scan the QR Code/link and see the codes). So, professors should use a computer or tablet, because the system was build around that scenario.

VI. Evaluation

To cover all its aspects, *I Am Here!* was evaluated in 3 separate ways: a Classroom Usage Evaluation (VI-A), an User Evaluation (VI-B) and a Device Fingerprinting Evaluation (VI-C). The evaluation served not only as a case study and to validate the system, but also to identify problems that should be corrected throughout the system's usability.

A. Classroom Usage

During the second semester of the school year 2018/2019, *I Am Here!* was implemented and used in *Multimedia Content Production* course in Técnico Lisboa. From the students' perspective, the system was evaluated according to its Effectiveness, (VI-A1) and Efficiency and Error Rate (VI-A2).

1) Effectiveness: To understand if students did check their attendance on the right time, it is required to calculate the effectiveness of the system. For this, in each class, during the attendance taking, the number of students was manually counted. To calculate this metric, it was used its respectively formula:

$$\frac{N_{registered}}{N_{present}} * 100 \quad (1)$$

Where `N_registered` is the number of attendances registered by the system, and `N_present` is the number of students in class.

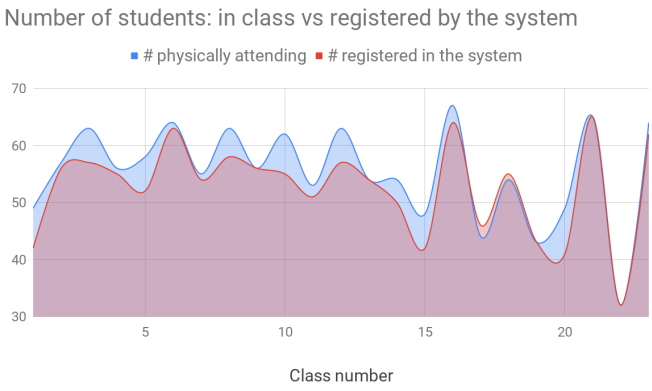

Figure 2. Number of students in class: physically and the ones registered by the system.

In Figure 2, the number of students physically in class (in red) and the number registered by the system (in blue) is represented for each class. Both numbers did not varied much. The biggest value was 8, and it only happened once, in class 20 - in this class, the internet was slow and students did not complete their attendance tacking (they did it at the end of the class). In fact, during class, some students just did not check their attendance - probably because they did not know that the course had a bonus for attendances. By the end of the system's implementation in the course, due to the reason mentioned before, it had an average of **95,05%** of effectiveness.

2) Efficiency and Error Rate: The second metric, the efficiency, serves to evaluate the number of codes the students inputted to check their attendance. Since this is related to the error rate (the number of incorrect codes inserted by the students) - if the error rate is high, the efficiency is lower

[4]https://regex101.com visited on June 23, 2019
[5]https://w3schools.com/w3js/w3js_sort.asp visited on June 23, 2019
[6]https://w3schools.com/tags/tag_meta.asp visited on July 4, 2019
[7]https://w3schools.com/css/css_rwd_viewport.asp visited on June 28, 2019

- both metrics are evaluated together. The efficiency formula will be calculated based on its formula:

$$\frac{N_{expected}}{N_{real}} \quad (2)$$

Where `N_real` is the number of inserted codes registered by the system, and `N_expected` is the number of inserted codes expected (in all classes, this number was always 3). For the error rate, the number of incorrect codes will be registered for each class.

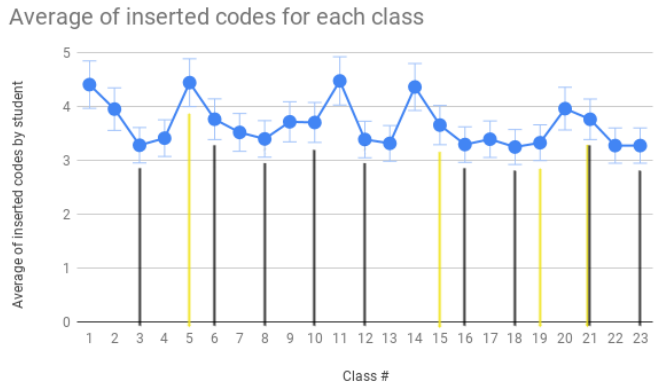

Figure 3. Average of inserted codes per student for each class (each dark line marks the classes with evaluation, and yellow the ones where students were a period of time without using the system).

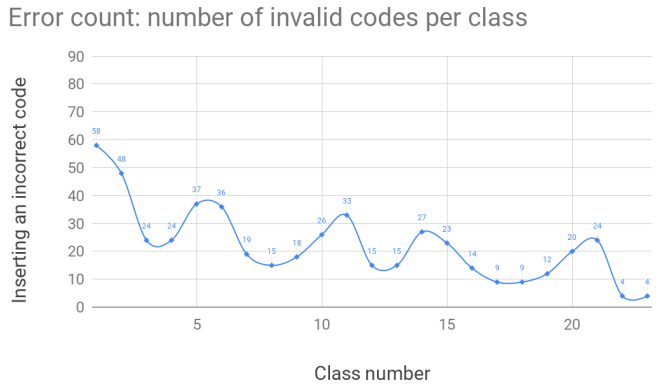

Figure 4. Error rate: number of incorrect codes inserted per class.

The average of inserted codes for each class is visually displayed in Figure 3 - the dark lines marks the classes with evaluation, and yellow the ones where the students were a period of time without using the system (due to holidays). The system ended up with an average efficiency of **82,80%**. In Figure 4, it is represented the number of incorrect codes inserted per class. In the first class, as it was the first interaction with *I Am Here!*, it was when more incorrect codes were inserted (a total of 58). As both metrics are related, when the average number of codes is higher, so is the number of incorrect codes.

Looking at Figure 3, in some classes (for example, class number 3), some students did insert fewer than the number of consecutive codes required (that was 3). After a few days, those students checked that they did not have their number on the attendance list of those classes. It was found that some students did not understand that it was required to insert 3 consecutive codes. After explaining (again), students started inserting the right number of codes, and this problem stopped happening. In other cases, it happened the opposite: students entered way more codes than what was expected. This happens when students do not enter the correct code, and have to enter more codes to check their attendance. As of the fifteenth class, the average number of inserted codes dropped to less equal than 4, given that students were getting more and more used to the system.

Because the course where the system was implemented had an evaluation every Monday and there were some students that only attended those classes, it was expected the peaks of Figure 4 illustrate that, but it was not the case. The first peak (class 5), was after a week without using the system - which is explained, because students had only four classes with the system, and were not yet used to it. But the following peaks (class 11 and 14) are not related with evaluations or *skipped* classes. These could be justified by the *difficulty* of the codes displayed - since codes are random, in some cases the code can be hard to memorize (in 10 seconds), so it increases the number of incorrect codes inserted.

On the seventeenth class, a countdown bar (representing the time remaining to input a code) was implemented for the first time. In that class, the number of incorrect codes was very low (only 9). The following peak (in class 20 and 21), it is suspected that, because there was no class on a Wednesday, students were one week without using the system, so, in the next class (number 20), there were an increase in the incorrect codes. Also, the classes where the minimum number of inserted codes is below 3 (that is the minimum required), are the cases where a student entered late in the class and did not have time to insert the number of consecutive codes required (because the professor closed the attendance session before they could finish). After the first three classes, 31 (out of the 71) students were already used to the system, inserting only 3 to 4 codes per class.

B. User Evaluation

To validate the solution, 20 users (comfortable around computers) were asked to perform 7 tasks, meant to cover most of the functionalities provided by the system. Before the tasks, a brief explanation of the system was done, and users had 5 minutes getting to know the system before starting the tasks.

- **Task 1**: Create an attendance session, for class number 5, entitled ”Invited Lecture - John Smith”, with type numbers, length 8, 10 seconds for each code, 2 consecutive codes and only enrolled students can access.
- **Task 2**: Check if student ”ist194044” the students that attended the class number 4 of Multimedia Content

Production, on the Alameda's shift, lectured by professor Daniel Gonçalves.

- **Task 3**: Get the permanent link of Alameda's shift attendances of Multimedia Content Production course.
- **Task 4**: Add the student *ist182083* to class number 3, in Alameda's shift, of Multimedia Content Production, as late.
- **Task 5**: View the students of Multimedia Content Production course.
- **Task 6**: Create the course "Linear Algebra", with "LA1819" as the ID and "2º Semestre 2018/2019" as the academic term.
- **Task 7**: Insert a professor to Multimedia Content Production course, with "ist182083" as the ID and "Michael" as the name.

In each task, the time to complete it, number of errors, number of tasks completed correctly (and incorrectly) was collected. For each tasks, a 5 minute limit was given to complete it. In the end of each task, a Single Ease Question (SEQ) was asked: *Overall, how difficult or easy did you find this task?*, were the user should answer in a scale between 1 (Very Difficult) and 7 (Very Easy). When all tasks were finished, a System Usability Scale (SUS) (with two extra questions added) was asked.

1) Results: Users that tested the system averaged 27 years old (8 women and 12 men). Neither one of the 20 were enrolled in *MCP* course or connected to the system whatsoever. All tasks were completed correctly.

*a) **Time to complete the task**:* Figure 5 shows a box plot with the time required to complete each task. Using one-way ANOVA, it determined that there was a statistically significant different ($F(6,133) = 18.92$, $p = .001$). A Tukey post hoc test revealed that the time to complete tasks 5 and 7 was statistically significantly higher, when compared with the remaining tasks. These tasks (and also task 4) were the ones where users took more time to complete.

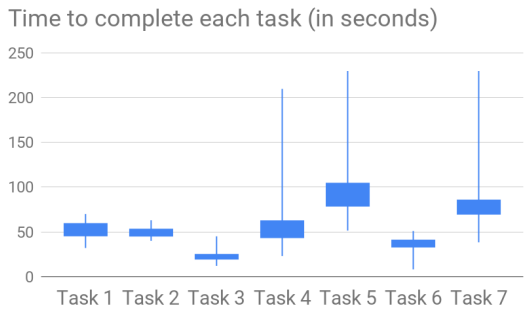

Figure 5. Time to complete each task.

*b) **Errors during each task**:* The tasks that took more time (tasks 4, 5 and 7) were also the ones that had higher errors by the users (Figure 6). Every time a user selected an incorrect button, an error was counted. For example, task 4 (*Add a student to a class as late.*), the input button was at the bottom of the page, requiring the user to scroll all the way down in order to be found/pressed. Having large tables and because users did not explore the system enough, they did find it harder to locate the input, and consequently took more time which means had more errors (users did click in almost all the buttons until finding the right one). The remaining tasks, besides some exceptions, had an average around 0 errors, which makes sense, since these were tasks that users did complete quicker.

Figure 6. Errors during each task.

*c) **User Ratings and Comments**:* Besides the usage metrics, as explained, users were asked questions to help understand what they thought about *I Am Here!*'s usability and all around usage. For the SUS (*Overall, how difficult or easy did you find this task?*), Figure 7 shows a box plot with the resulting answers, from 1 (Very Hard) to 7 (Very Easy).

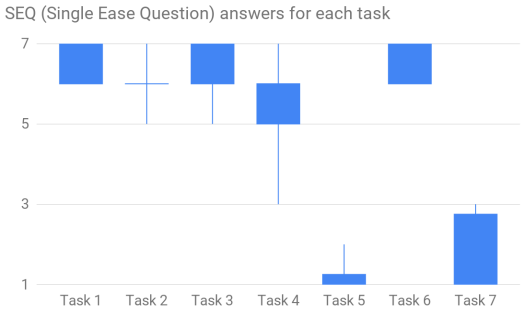

Figure 7. SEQ (Single Ease Question) answers for each task, where users had to answer in a scale of 1 (Very Hard) to 7 (Very Easy).

From Figure 7 it is possible to conclude that users did find tasks 5 and 7 very hard to complete. For task 5 (*View the students of Multimedia Content Production course*), the feedback was that the tab that was supposed to be selected was not evident. Since all the *main* buttons are in the center of the screen (and this tab was in the top left corner), as seen, users took more time, did more errors and that let to this task being hard to accomplish. When the task was accomplished, users did say that, after-all, if they had noticed that tab, it would have been faster. About task 7 (*Insert a professor to Multimedia Content Production course, with "ist182083" as the ID and "Michael" as the name*), users said the place and button's icon to go to the professor's edit menu was not evident.

Since users had a couple of minutes to get used to the system, the remaining tasks were stated as simple (for example, task 3), where users did take a short amount of time and had less errors, comparing to others that required more attention (task 7, for example). These is also confirmed with the answers from SEQ (as per Figure 7).

The overall SUS score is **89.25**, which means that the system is *excellent* (every system with a 80.3 or higher SUS score is graded as *excellent*. The odd numbered question were the ones were the intended responses were closer to 1 (Strongly Disagree), and for the even number questions the opposite. For example, "I think that I would like to use this system frequently", which averaged a score of 4,75. In "I found the system unnecessarily complex." (Question 2), obtained an average score of 1,4, which, given the question, means that uses did not find the system complex.

For the two *extra* questions added, "I think the functionalities of the system are enough for a professor" (Question 11) users agreed, having an average score of 4,86 (the question with higher score). Finally, the "I wanted to do something but I did not find the way to do it" (Question 12) yielded a score of 2,25, meaning that, even though users took more time in some tasks, they managed to complete it anyways.

C. Device Fingerprinting Evaluation

In order to catch fraud attendances, a device fingerprint evaluation was also done. A set of scenarios were tested, meant to cover most of the more common attempts of fraud by the students. The scenarios were:

- **Scenario 1**: A student takes an attendance for him/her and a classmate using the same device and browser for both attendance checking.
- **Scenario 2**: A student takes an attendance for him/her and a classmate using the same device and a different browser for both attendance checking.
- **Scenario 3**: A student uses a different device than usual to record the attendance (ex: a colleague is using two different device to record two distinct attendances).

For each scenario, it will only be taken into account if the system detects the fraud or not, and a brief explanation why.

1) Results and Discussion: The results and discussion are discussed for each one of the scenarios separately.

a) Scenario 1: In this case, if a student takes two attendances in the same device and browser, the device's fingerprint will be exactly the same (since both attendances do not have any attribute to distinguish them). Having no attribute to distinguish them, the system detects this case, and students are flagged.

b) Scenario 2: Changing the browser does change some attributes in a device fingerprint. For example, the user agent. This indicates different aspects about a device, and it differs from browser to browser. Using, for example, Google Chrome and Firefox, the User Agent is, for the same device, respectively:

Google Chrome: `Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36`

Firefox: `Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:59.0) Gecko/20100101 Firefox/59.0`

Since it does change using a different browser, and it is also possible to manually change it, it is good practise to not use the user agent as an important attribute when evaluating a student's device's fingerprint. But, other attributes do stay the same, are browser independent. For example, the IP Address (in this case, it is used the IP Address, because it it unique, and other attributes, for example `Content Language` or `Platform` can be the same for distinct devices). Having the same IP Address, the system can detect this case, and students are flagged.

c) Scenario 3: For each class, the student's device fingerprint information is stored. Except for the first class, and taking into account students use the same device and browser to take the attendance (people usually do not change their browser of choice), the current fingerprint is compared with the previous one. Besides the IP Address, that will likely never be the same, the remaining attributes stay the same. In this case, if a student does an attendance for them or a colleague, if it is different from the previous one, the system *detects* and the student is flagged.

D. Discussion

Overall, the results of the usability user tests show that *I Am Here!* is understandable and userfriendly. The main part, which is to create an attendance, analyze the students that took the attendance and the insertion of codes was found easy by the users. This is shown by the low amount of time and errors to complete those tasks. For the more "complex" task (i.e., the tasks were not hard to do, but were not visible to the naked eye for everyone) users took more time and therefore had more errors.

From the questionnaire, similar results were obtained; when asked "I think the functionalities of the system are enough for a professor", it obtained an averaged score of 4,85, meaning that the system fulfilled all the professor's needs. Users also felt that the system was easy to use (average score of 4,65) and that they did not need to learn a lot of things before using the system (Question 10, which average score of 1,85). This shows that *I Am Here!*'s main goal (to simplify the professor's job to check the students attendance in a classroom) was accomplished by implementing clear and simple features.

The same can be said from the student's perspective, with a high efficiency (82,31%) and with (almost) every student taking their attendance (giving a high effectiveness), it is possible to conclude that, from this perspective, it was also accomplished a system that does take the students' attendance in a automated way, without taking too much time of the class.

Having the Device Fingerprint was also important to uncover the most common fraud cases that can happen. The more common scenario - when a student uses the same device to

take distinct attendances - is detected by the system. Having a short time to insert each code and device fingerprint, *I Am Here!* can be described as a robust automated attendance control system.

VII. Conclusion

Checking the students attendance in a classroom can be very important for a professor, from understanding which type of subjects are more appealing to students to simply having a record of how many students are in class. In *I Am Here!*, it was possible to implement an attendance system that not only checks the students attendance but, among other things, it is easy to use, does not require (extra) hardware and it is possible to export that information (to use in other context). Is was possible to evaluate the system in a real life scenario - in class. This evaluation was very helpful to improve the system to (almost) its best. During the system usability it was possible to discover details and events that could only be discovered during class.

From now on, *I Am Here!* can be used by any professor in Técnico Lisboa. The code is public on GitHub, so anyone can use it and/or pick up the work and expand it [8].

A. Future Work

Each year, students will manage to have new ways to cheat the system, so exploring the Device Fingerprinting should be the next important step. Then, expanding the system to other faculties (i.e., changing the way of login, to something more universal, such as Google or Facebook), as it is important to analyze if different students (from different courses) interact the same way with the system.

VIII. Acknowledgments

To my supervisor, Daniel Jorge Viegas Gonçalves for his exemplary guidance, monitoring, constant encouragement throughout the course of this project. Thank you.

References

[1] A. Kassem, M. Hamad, Z. Chalhoub and S. El Dahdaah, 'An RFID attendance and monitoring system for university applications', *2010 IEEE International Conference on Electronics, Circuits, and Systems, ICECS 2010 - Proceedings*, pp. 851–854, 2010. DOI: 10.1109/ICECS.2010.5724646.

[2] M. I. Moksin and N. M. Yasin, 'The implementation of wireless student attendance system in an examination procedure', *2009 International Association of Computer Science and Information Technology - Spring Conference, IACSIT-SC 2009*, pp. 174–177, 2009. DOI: 10.1109/IACSIT-SC.2009.130.

[3] M. Kassim, H. Mazlan, N. Zaini and M. K. Salleh, 'Web-based student attendance system using RFID technology', in *Proceedings - 2012 IEEE Control and System Graduate Research Colloquium, ICSGRC 2012*, 2012, pp. 213–218, ISBN: 9781467320368. DOI: 10.1109/ICSGRC.2012.6287164.

[4] A. Charity, K. Okokpujie and N.-o. Etinosa, 'A Bimodal Biometric Student Attendance System', *IEEE 3rd International Conference on Electro-Technology for National Development (NIGERCON)*, pp. 464–471, 2017.

[5] S. Lukas, A. R. Mitra, R. I. Desanti and D. Krisnadi, 'Student attendance system in classroom using face recognition technique', *2016 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 1032–1035, 2016. DOI: 10.1109/ICTC.2016.7763360.

[6] V. Tunbunheng, 'Automatic attendance system for late student using speech recognition corresponding with google forms and sheets', *Ubi-Media 2017 - Proceedings of the 10th International Conference on Ubi-Media Computing and Workshops with the 4th International Workshop on Advanced E-Learning and the 1st International Workshop on Multimedia and IoT: Networks, Systems and Applications*, pp. 4–7, 2017. DOI: 10.1109/UMEDIA.2017.8074119.

[7] B. A. Talip, 'MOBILE ATTENDANCE SYSTEM USING QR CODES TECHNOLOGY', vol. 3, no. 1, pp. 1–3, 2018.

[8] R. Asa'D and C. Gunn, 'Improving problem solving skills in introductory physics using Kahoot!', *Physics Education*, vol. 53, no. 5, 2018, ISSN: 13616552. DOI: 10.1088/1361-6552/aacade.

[8] https://github.com/matildepgrn/I_Am_Here