I Am Here!

Automating Attendance Control

Matilde Paraíso Guerreiro Rosa do Nascimento

Thesis to obtain the Master of Science Degree in
Information Systems and Computer Engineering

Supervisor: Prof. Daniel Jorge Viegas Gonçalves

Examination Committee

Chairperson: Prof. Luís Manuel Antunes Veiga
Supervisor: Prof. Daniel Jorge Viegas Gonçalves
Member of the Committee: Prof. Alfredo Manuel Dos Santos Ferreira Júnior

July 2019
Acknowledgments

I would like to thank my parents for their friendship, encouragement and caring over all these years, for always being there for me and without whom this project would not be possible.

To my siblings, Ana and Diogo, I would like to thank for their constant support and wisdom. They will always be an example for me.

To my second family, Tuna Mista do Instituto Superior Tecnico (TMIST), my sincere thanks for all the nights of “foly” that I had with you. I will always have you in my heart.

To Luís Silva, for all his love, support and patient that kept me going through this adventure.

To my friends, for keeping me sane, and for all the advises and laughs we had together.

To my supervisor, Daniel Jorge Viegas Gonçalves for his exemplary guidance, monitoring, constant encouragement throughout the course of this thesis, for all the (extra) meetings and for trusting me in the first place, when accepting me as his student.

This work was supported by national funds through Fundação para a Ciência e Tecnologia (FCT) with reference UID/CEC/50021/2019 and project GameCourse (PTDC/CCI-CIF/30754/2017).

Thank you.
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.

For this solution, it was developed a web-system using JavaScript, HTML5, CSS and MySQL. As a case study and form of validation, the system was deployed in the Multimedia Content Production course, where 116 students were enrolled, in Técnico Lisboa. Throughout the course, the system was submitted to three evaluations. A Classroom Usage Evaluation where, from the students’ perspective, the system was evaluated according to its effectiveness (95,05%) and efficiency (82,80%). An User Evaluation, with 89,25 as the SUS score, it was possible to improve some functionalities. And also a Device Fingerprint Evaluation, where three of the most common fraud cases were tested and proven to be detected by the system.

Keywords

students attendance system; automated attendance system; device fingerprinting.
Resumo

A marcação de presenças dos alunos foi sempre um problema para o professor, não só pelo tempo que consome em aula, como a complicação que é ter um sistema que garanta que o aluno está fisicamente presente. *I Am Here!* tem como objetivo ser um sistema inovador, tendo como base certos requisitos, nomeadamente: ser barato, portátil e resistente a fraude, recorrendo à técnica da Impressão Digital do Dispositivo.

Para esta solução, foi feito um website em JavaScript, HTML5, CSS e, para base de dados, MySQL. Para testar o sistema, este foi implementado na unidade curricular Produção de Conteúdos Multimédia, onde 116 alunos estiveram inscritos, no Técnico Lisboa. O sistema foi avaliado de três maneiras distintas. Uma avaliação em aula, onde foi possível avaliar, da perspectiva do aluno, a eficiência (95,05%) e eficácia (82,80%) do sistema. Uma avaliação com utilizadores, onde se obteve 89,25 na escala de usabilidade do sistema e foi possível fazer melhorias ao sistema da perspectiva do professor. E por último, uma análise à Impressão Digital do Dispositivo, onde se provou que, dos três tipos de fraude mais comuns, o sistema consegue detetar.

Palavras Chave

sistema de marcação de presenças; sistema de marcação de presenças automático; impressão digital do dispositivo.
# Contents

Acknowledgements ................................................. i  
Abstract ................................................................... iii  
Resumo ....................................................................... v  
List of Figures ........................................................ xi  
List of Tables .......................................................... xiii  

1 Introduction 1  
1.1 Objectives ......................................................... 4  
1.2 Contributions ...................................................... 4  
1.3 Document outline ................................................. 5  

2 State Of The Art 7  
2.1 Manual Attendance System (MAS) ......................... 9  
2.2 Radio-Frequency Identification (RFID) ................... 9  
2.2.1 RFID in attendance systems ............................. 10  
2.3 Biometrics ......................................................... 10  
2.3.1 Biometrics in attendance systems ..................... 10  
2.4 Mobile devices and GPS location ......................... 11  
2.4.1 Mobile devices in attendance systems ............... 11  
2.4.1.1 GPS location in attendance system ............. 12  
2.5 Software-only systems ........................................ 12  
2.5.1 Quick Response (QR) Code ............................. 13  
2.5.2 Kahoot! ......................................................... 13  
2.5.3 Top Hat .......................................................... 13  
2.6 Device fingerprinting .......................................... 14  
2.7 Discussion ......................................................... 14  

3 Proposed Solution 17  
3.1 Requirements ...................................................... 19  
3.2 Approach .......................................................... 20
# Implementation

## I Am Here!

### FenixEdu

- Create course
- Create a shift
- Insert professor(s)
- Associate a professor to a shift
- Insert/Update student(s)
- Start a new attendance
- Insert a previous attendance
- Attendance history
- Edit class information
- Export attendances
- Detecting fraudulent attendances
- Student’s information
- Course’s attendance history

### Student’s perspective

## Architecture

## Implementation

### Server

- Authentication
- FenixEdu API
- Services
- Attendance Workflow
- A – Synchronization

### Database

### Scalability
# List of Figures

3.1 Professor’s perspective of the attendance checking. ........................................ 21
3.2 Student’s perspective of the attendance checking. ........................................ 21
3.3 Example of a test; letters and numbers as a code type, with a 7 characters length. 22
3.4 Box plot with the time taken during the testing phase. .................................. 23
3.5 Error count during the testing phase. ............................................................. 24
3.6 Answers from the questions during the testing phase. .................................. 25
4.1 *I Am Here!* initial page. ................................................................................. 29
4.2 *I Am Here!* course’s page. ............................................................................. 30
4.3 List of the professor’s courses that are not selected. ..................................... 30
4.4 *I Am Here!* shift’s page. ................................................................................ 31
4.5 Edit shift(s) page. ............................................................................................ 31
4.6 Add/remove a professor functionality. .............................................................. 32
4.7 Import students functionality. .......................................................................... 33
4.8 *I Am Here!* attendance’s page. ..................................................................... 35
4.9 Attendance personalization page. ..................................................................... 36
4.10 Attendance links page. ...................................................................................... 37
4.11 Attendance checking page. ................................................................................ 38
4.12 Class view. ....................................................................................................... 39
4.13 Brief summary of a student’s fingerprint. ......................................................... 39
4.14 When the student inserts a correct code. ......................................................... 40
4.15 When the student inserts an invalid code. ....................................................... 40
4.16 *I Am Here!* architecture. ............................................................................ 41
4.17 *I Am Here!* server files organization. ......................................................... 43
4.18 *I Am Here!* database schema. ..................................................................... 47
4.19 Part of the front-end file organization. ............................................................. 49
5.1 MCP’s schedule during the school year 2018/2019. ................................. 55
5.2 Number of students in class: physically and the ones registered by the system. .... 56
5.3 Difference: students in class and the ones registered by the system. ................. 57
5.4 Average of inserted codes per student for each class. ............................... 58
5.5 Efficiency (in %) for each class. ................................................................. 59
5.6 Time (in minutes) the attendance system was running for each class. ............... 59
5.7 Number of codes insert per student in each class. .................................... 60
5.8 Number of students that became comfortable using the system in each class. .... 60
5.9 Error rate: number of incorrect codes inserted per class. ........................... 61
5.10 Time to complete each task. ................................................................. 64
5.11 Errors during each task. ................................................................. 65
5.12 SEQ (Single Ease Question) answers for each task. ............................. 65
5.13 System Usability Scale (SUS) average answers. ................................. 66
List of Tables

2.1 Comparison between different student’s attendance systems. 15

5.1 Time (in seconds) to complete statistics for each task. 64
5.2 Errors statistics for each task. 64

A.1 *I Am Here!* effectiveness results. 80
A.2 *I Am Here!* efficiency results. 81
A.3 *I Am Here!* error rate results. 82

D.1 System Usability Scale answers by each user. 88

E.1 Time (in seconds) to complete each task: minimum value, lower quartile, upper quartile and maximum. 89
E.2 Errors during each task: minimum value, lower quartile, upper quartile and maximum. 90
E.3 Single Ease Question (SEQ) answers for each task: minimum value, lower quartile, upper quartile and maximum. 90
1

Introduction

Contents

1.1 Objectives ................................................. 4
1.2 Contributions ............................................. 4
1.3 Document outline ....................................... 5
There are many situations where attendance tracking is necessary, so it is possible to distinguish who arrives on time and who arrives late. 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 [1]. Both of these systems are vulnerable and not scalable for large numbers of students, being too time consuming (the first more than the second). The second one can also be easily trespassed - simply by having one student signing for a colleague (or more). Both also require some type of human oversight ¹ and, afterwards, takes a good amount of time to transfer the information to a computer. 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), this type of system is neither reliable nor effective in any way.

Some other types of attendance systems also exist, but do require (extra) hardware, i.e., hardware that is external from what a professor usually carries (that usually is a computer) - for example, a RFID (Radio-Frequency Identification) reader, that scans the student's ID, or a Face Recognition system [2] [3]. This can be a disadvantage, because it requires maintenance (and this usually means someone external, which means more time consuming) and can be hard to understand and use by a regular professor (that may not be used to it), as well as the possible cost to acquire and trouble to move between classrooms. Also, if a student does not allow to take a picture of them, the face recognition system can not be used. As it is meant to be used on a daily basis and by any professor, these type of systems can also not be the best suited.

With increased importance given to attend classes, such as the knowledge that is acquired (or when there is an 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.

Attendance systems are a prime example of an area where a lot of work needs to be done so that any professor can check their students' attendance more easily, without needing to use expensive tools or consume time outside the classroom and assuring it is fraud resilient. As this is a daily routine in a professor's job, giving that we are living in a technological era, it is fundamental to have an accurate system that manages to fulfill all the professors needs.

1.1 Objectives

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. In this context, as opposed to *automatic* - when the action happens by itself, without human’s manipulation - an *automated* system does not have much interaction from the user’s perspective - the action is focused on a machine (in this case, a computer) and not the human \(^2\).

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 - this tool allows to create a (unique) fingerprint for each students’ devices. With this, it is possible to distinguish each student’s device fingerprint or, in a fraud case, disclose two (or more) even fingerprints.

To test the system, usability user tests were done for both students’ and professor’s side, as well as a device fingerprinting evaluation. Usability user tests allows us to infer whether *I Am Here!* reached the goals of being simple and effective in an everyday class. To assess how users interact with the system, 20 participants were asked to complete a set of tasks. For the device fingerprint analysis, a set of scenarios were made - to cover the most common fraud cases - to assess if the system could catch those cases. The results show that *I Am Here!* is simple to use and understand, that can also catch the most common fraud attempts. How it was implemented and evaluated, as well as a detailed explanation of the interface, is described in the following sections.

1.2 Contributions

We created a working solution to check the student’s attendance in a college class. *I Am Here!* is, among other things, a simple, costless, portable and fraud-resistant system that eases one of the professor’s job - to know which students are in the classroom. Using Device Fingerprint analysis, it can also detect the more common fraud attempts in this field.

To understand how the students would interact with the repeated insertion of codes before the system went live, we performed a study on repeated code-entry, where different code sizes, types and repetitions were assessed. This may inform similarly-based approaches and is described in chapter Proposed Solution (section 3).

\(^2\)https://englishforums.com/English/AutomaticVsAutomated/bczlkx/post.htm visited on June 29, 2019
Using HTTPS (Hyper Text Transfer Protocol Secure) and a secure login service, I Am Here! allows professors to create and personalize courses and its aspects (shifts, students and attendances). This allows to implement the system in any college.

A paper, describing the work done and its evaluation during this thesis, was submitted to the ICGI 2019 – International Conference on Graphics and Interaction 2019.

Throughout the system’s implementation, all versions and updates were being stored in a GitHub project\(^3\), as it is the most known free to use software development version controller.

1.3 Document outline

This document is divided into 6 chapters and several appendices:

- **Chapter 1 Introduction**: Overview of the thesis’ scope.
- **Chapter 2 State Of The Art**: A review of previous scientific work of importance to the research.
- **Chapter 3 Proposed Solution**: Explanation of the proposed system.
- **Chapter 4 Implementation**: Detailed information about the thesis implementation and its functionalities.
- **Chapter 5 Evaluation**: Description and interpretation of the evaluation’s results.
- **Chapter 6 Conclusion**: Final remarks.
- **Appendix A Classroom Usage Evaluation Data**: I Am Here!'s evaluation data: effectiveness, efficiency and error rate results.
- **Appendix B System Usability Scale (SUS)**: System Usability Scale (SUS) questionnaire.
- **Appendix C Single Ease Question (SEQ)**: Single Ease Question (SEQ) questionnaire.
- **Appendix D Single Ease Question (SEQ) results**: Single Ease Question (SEQ) questionnaire results.
- **Appendix E User Evaluation Evaluation Data**: Time, Errors and SEQ results from the User’s Evaluation.
- **Appendix F Step by Step**: how to update HTTPS certificates: How to update HTTPS certificates for I Am Here!'s website.

\(^3\)https://github.com/matildepgrn/I_Am_Here
State Of The Art

Contents

2.1 Manual Attendance System (MAS) .................................................. 9
2.2 Radio-Frequency Identification (RFID) .......................................... 9
2.3 Biometrics .................................................................................. 10
2.4 Mobile devices and GPS location ................................................. 11
2.5 Software-only systems ............................................................... 12
2.6 Device fingerprinting ................................................................. 14
2.7 Discussion ................................................................................ 14
In this section, it is presented all the theoretical research on the relevant topics for the project. Given the scope, it is useful to research different types of Automated Attendance Systems (AAS). Because the solution aims to eliminate time unnecessarily consumed by the students while having to manually fill an attendance sheet, the research was mostly around systems with the same mindset.

2.1 Manual Attendance System (MAS)

The most common means of attendance system is the manual one [1]. This system normally works with an attendance sheet that has the name (and in some cases, the number) of the students (ordered alphabetically by name, for example), the date and time of the class, and the name of the course. This sheet is passed through the class, an it is supposed to be signed only by the students that are actually attending the class.

This approach presents a variety of disadvantages. The most notorious one is that anyone can sign for anyone that is not inside the class. In cases with a lot of students inside the room, is it not possible for the professor to keep track who is signing more than one name in the sheet. Another problem is afterwards, when the professor has to manually check all the signatures. Even if the class does not have a lot of students (for example, a class with 30 students), this process can be very time consuming. Not only because of the repeated action to check the signature and mark it in a virtual spreadsheet, but because the professor usually does not teach a single course and therefore needs to repeat this for other courses, as well. In a scenario where a professor has at least two courses, there are at least four classes at the end of the week to verify. A semester has around 14 weeks, so in the end, there are at least 56 sheets to manually verify. Another problem is the amount of paper that is used for this system - it is expensive and takes a lot of space to store.

This is a very slow process, that could be easily implemented in a more automated manner, without wasting paper and physical space.

2.2 Radio-Frequency Identification (RFID)

Radio-Frequency Identification, more commonly called RFID, is a technology that was invented around the 1940s as a means for remotely identifying military aircraft [2] [4]. It was also used for industrial and commercial purposes by the 1980s, but nowadays, because it is a wireless technique for identifying physical objects remotely, where the wave frequency of their tags stands between 3 KHz and 300 GHz, although the system overall (with all the necessary components) is still expensive, this technology can replace other concepts that are used in our daily-bases, such as bar code and magnetic cards, because of its various advantages [5] [6].
RFID systems require two main components: RFID readers and tags, and an application or software package running on a computer [1].

2.2.1 RFID in attendance systems

RFID has been used as an student attendance checking system in several teaching facilities. It normally has a passive reader in the entrance of the classroom, and students need to flash their cards to check their attendance [1]. It is also possible to check online, for example in the university's website, how many classes a student has and has not attended.

Although this type of system does prevent some of the problems of manual attendance checking system (the fact that is automated), this one does not prevent from bringing another student’s card and to flash it as well, and does not have a backup solution when a student forgets his/her card. It also requires extra hardware; some faculties do not have RFID readers in every room, and in a case where a class needs to be taken in a different room, it could cost time of the class. Also, not only the maintenance and repair of this type of equipment can be really expensive, but this system, in a class with not just a few students, can create a queue for who is waiting to flash their card (which can delay the beginning of the class).

2.3 Biometrics

Biometrics is a type of system that extract key features of human characteristics. It does the identification and verification given physical and behavioral traits, such as fingerprints, irises, retinal patterns, voice and key point features of the human’s face - nose, mouth, edges, eyes and other characteristics [7].

In the scope of automated attendance systems, this technique does not require that students bring extra hardware to take the attendance, but it must exist in the faculty, and it also needs to be maintained.

2.3.1 Biometrics in attendance systems

Biometrics can be used in many ways for attendance checking system in classrooms. In Thailand, for a Computer Architecture class in the Faculty of Engineering and Technology, it was implemented the combination between Google Forms and a Speech Recognition system [8]. 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.
Other option is to use fingerprint; G. Talaviya et. al. created a system where, in the first 20 minutes of the class, students can mark their presence by simply putting their finger in a fingerprint sensor, that can be moved dynamically and wirelessly by the students around the classroom [9]. After scanning their fingerprint, students can check their attendance being registered just by looking at the screen projected on the wall. In this system, an updated attendance email is send to the professor and the students, so they can be informed about the student’s attendance.

It is also possible to combine biometric and face recognition for an extra layer of authentication. In Covenant University, in Nigeria, it was implemented a bimodal biometric automated attendance system, using the combination of facial recognition and fingerprint from the students [3]. The students enter the classroom and have to, one at a time, individually, use the professor’s computer to take ten different pictures (with different postures) and capture his/hers fingerprint four times in a fingerprint reader. With this double authentication, it is possible to erase the problems with just one biometric system.

Other functionalities can be used with face recognition. In Slovakia, besides the attendance tracking system, it was used to check if a room was empty or not - this is a good use of this technique, in a case where a class is dismissed earlier, for example [10].

2.4 Mobile devices and GPS location

It is possible to carry a lot of information and features with a mobile phone - one of them can be to check attendance in a classroom, with passwords or current location, for example.

This type of attendance checking system has the advantage of not requiring the students, faculty and professors to carry extra hardware because nowadays everyone carries a smart-phone in their pocket.

2.4.1 Mobile devices in attendance systems

For an attendance checking system, a basic implementation can be done with mobile devices. As an example, in Khulna University of Engineering & Technology in Bangladesh it was created a system that uses an Android app where students can register their attendance [11].

A more complex system was implemented in a Japanese university: an attendance tracking system using a Bluetooth Low Energy (BLE) beacon and Android devices [12]. In class, the professor turns on the BLE beacon to transmit a magic number to the Android devices inside the room. The students then run the application and scan their cards on a NFC reader. The Android devices receive the magic number and then send it with the scanned ID and name together, to the server. This system not only registers the attendance of students, but it also reduces the time for attendance registration, using multiple devices in parallel; it also eliminates false attempts from students outside the classroom, because of the BLE
beacon device - the signal covers the entire classroom (about 30 meters) and it is blocked by the wall, so no one outside can receive the magic number.

It is also possible to use a mobile phone as a security token for authentication [13]. Hallsteinsen et al implemented a One-Time Password (OTP) where, to mark attendance, the students had to complete a challenge that is sent to their mobile phone after logging in. Also using OTP, in American University of Sharjah, exists a system that generates a password with factors that are known by both entities - user and server [14]. If the password matches, the user is authenticated. If this password expires, an SMS-based mechanism is also implemented so it is possible to retrieve the password to connect to the server.

Also, in Técnico Lisboa, using a small USB powered router wireless access point, IocDev was developed as a means to check the student’s attendance [15]. In this system, students need a way to access the internet (via Wifi), and then just need to provide their credentials, the class pin (that the professor writes on the board) and the current shift - besides the first class, where it is required to create an account. The attendance is only valid if it is within the class schedule, and it is restricted to one attendance per device. Also, the attendance is only valid if the student’s device can connect to the Wifi access point, that is the student is inside the class (yet, students that are very close to the classroom may be able to connect).

2.4.1.A GPS location in attendance system

Applying location in an attendance system was implemented in SMARTDOT [16]. This system requires the students to download an Android application and it calculates the distance between the student’s and the professor’s device. A more complex approach, also using location as an attendance system, was done in Beijing, in School of Software, where it was implemented an automated student attendance tracking system based on location (and voiceprint) [17]. For the location part, because GPS signal is too weak, it is used a service provided by Baidu Inc, which combines results from GPS, cellular network and Wi-Fi signals. In class, after the voiceprint verification, the distance between the student’s mobile phone and the professor’s computer is verified; if the distance is in range, the attendance is checked, otherwise, it is not.

2.5 Software-only systems

Also related to portable devices, there are some softwares 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. From various software-only systems, the more common ones are the Quick Response (QR) Code and Kahoot!, that
are explained in the following two sub-subsections. A not so common solution is Top Hat, that is explained in 2.5.3.

As opposed to the attendance systems that were previously described, this software-only systems not only do not require (extra) hardware, but also do not have the sole purpose to do the attendance checking, but can and commonly are used in that sense.

2.5.1 Quick Response (QR) Code

Quick Response Code, better known by QR code, is a technology that has been widely used by many industries [18]. 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.

2.5.2 Kahoot!

Kahoot! is a company born in March 2013 with the purpose of improving the education around the world [1]. 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 the end of the quiz, it is possible to check the top 3 players.

In University of Moratuwa, in a class to teach English as a Second Language (ESL), two types of vocabulary quizzes where made for two distinct groups of 15 students each: a pen and paper one, and another using Kahoot! [19]. After weeks of testing both quizzes, it was concluded that the group taking quizzes using Kahoot! had better results, as well as more attendance. Also, in United Arab Emirates, in an introductory physics class, 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) [20].

2.5.3 Top Hat

Also using a web browser, Top Hat is a good solution for a costless attendance system [2]. Requires the students to correctly submit a 4-digit code and to be present in the classroom. The presence is checked

---

by the geolocation and proximity to other students that are also submitting the code (it requires access to the device’s location to be allowed by the students).

2.6 Device fingerprinting

It is known that Internet of Things is becoming an essential part in the daily life of those who manipulate wireless technologies [21]. Looking on the bright side, this technologies can be connected anytime and anywhere, at a low cost. Yet, they also come with vulnerabilities, that can compromise confidentiality, integrity and authentication of data communication.

There is a process that can gather signatures and characteristics from a user called Device Fingerprinting. The basic idea is to gather device-specific information (such as mobile phone’s brand, screen’s dimension or the list of installed fonts) to form a signature - an identification - of a specific user - applying this not to a couple informations, but many, to thousands of people. A experiment made in 2010 showed that half a million users who participated, using Flash or Java, had a unique device fingerprint [22].

This technique can be used in attendance systems to validate if a student is indeed taking his/her attendance and to catch fraud cases - if a student suddenly changes its way of taking the attendance, or if the fingerprint matches exactly a colleague, this technique can flag the students.

2.7 Discussion

After researching different types of student attendance systems, there are advantages and disadvantages that were taken into account while developing the proposed solution, according with the Objectives in section 1.1. In Table 2.1, the previous mentioned students attendance system are compared with the features that the proposed system aims to achieve.

It is possible to verify that the majority of them does not use paper (only the Manual Attendance System (MAS)), but on the other hand do require extra hardware; more common was hardware that the faculty must have and maintain, such as an RFID reader, or a camera. This is a major disadvantage, because it can be expensive not only because of the material itself, but because it is required to exist in all the classrooms. And because of this, they are also not easily implemented; requires a technician to implement and keep the systems up to date and maintained. In large universities, this can be a problem and can compromise some classes (if the system is not available to be used, or needs to be fixed).

Only the Mobile devices and GPS location and Software-only systems require the students to carry an internet-connect device (such as a mobile phone or computer), offering no option for the students who do not. Also, in underground or indoors classroom, the GPS signal can be weak, which compromises the attendance checking.
Table 2.1: Comparison between different student’s attendance systems.

<table>
<thead>
<tr>
<th>Uses paper</th>
<th>MAS</th>
<th>RFID</th>
<th>Biometrics</th>
<th>Face Recognition</th>
<th>Mobile</th>
<th>Software only</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>(extra) Hardware</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Requires Internet-connected device</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Expensive</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Portable</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Efficient</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Time consuming</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Violates privacy concerns</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Can be cheated</td>
<td>✓</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>✓</td>
</tr>
<tr>
<td>Can be done after class starts</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Easily implemented</td>
<td>✓</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(afterwards) Available online</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Overall, the systems can be available online and are effective, as they get the job done - the attendance is checked for, at least, the students that are in the classroom. But, on the other hand, some can be time consuming. In classes with a lot of students, some systems (namely Manual Attendance System (MAS), Radio-Frequency Identification (RFID) and Biometrics) can create a queue of students, waiting to take their attendance, which can delay the beginning of the class.

All the systems allow the students that do not arrive on time, can also check their attendance. This is an advantage and can also be a disadvantage, because taking the attendance can disturb or, in some situations, stop the class (in a scenario where the professor is involved in the verification).

The Biometrics systems, although they have a high percentage of recognition/match, present privacy concerns for the students. For instance, if one student does not allow a picture of them to be taken during class, or does not want to scan their fingerprint, systems using this type of verification will not be effective.

With all the systems, only the Manual Attendance System (MAS), Radio-Frequency Identification (RFID) and the Software-only systems can be cheated. The first one, because it is possible to sign for another students without calling the professor’s attention; the second one, it is by simply flashing someone else’s card (in a class with a lot of students, or when the RFID reader is not close to the professor, it is impossible to keep track of how many cards a student is flashing). The third and final one, in systems where more than one student can take attendance in the same device, without GPS location, it is possible to take the attendance for more than one student.

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.
3 Proposed Solution

Contents

3.1 Requirements ................................................................. 19
3.2 Approach ................................................................. 20
3.3 Understanding Code Entry ............................................. 21
As described in the first section, the main problem with attendance systems is the time it consumes and its low efficiency to guarantee that students are in the classroom. After searching about the different type of attendance systems, measuring all the pros and cons, to implement I Am Here!, a list of Requirements was made, to define what were the main objectives of the system, and then what was the right Approach to do. Then, to better understand the feasibility of a code-entry based solution, a study dedicated to Understanding Code Entry was conducted, to better understand how the target audience (students) would interact with this type of system, before it went live.

3.1 Requirements

This system must check the student’s attendance in a faculty class, in a way that:

- **does not require extra-hardware** (besides what students and professors usually carry, such as a mobile phone, computer or tablet), which means that:
  - the professor does not have to carry anything between classes
  - the system will not cost money
  - it will be portable
- if a student **does not have a way to access the internet**, it is also possible to check their attendance;
- **is not time consuming** (for the professor and the students) before, during and after class;
- **is effective**, as it checks the attendance of the students that are in the class;
- **only students inside the classroom can check their attendance**, i.e., it should be hard to trick the system;
- **does not violate privacy concerns**;
- the student’s attendance can be made **available online** (so an updated attendance record is only accessible to students and professor);
- can **easily be implemented** in any course;
- **can distinguish between students arriving on time and being late**;
- **students that are not (officially) enrolled in the course** can still check their attendance.

This system assumes that the classroom has a projector (or a large TV) and that the professor has a computer available. If one of these assumptions does not happen, the system cannot be used.
3.2 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?*. As explained in Chapter 1, *automated* does not mean automatic, but that has little interaction from the user's perspective, and the main work is done by a machine (which, in this context, is a computer). To achieve the objectives listed in section 3.1 a set of characteristics and boundaries will be implemented.

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 (such as mobile phones and computer, which is what a university student usually carries), 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 the code for another person or even send it.

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 and could be asked by the professor afterwards - to disguise a possible attempt of a student taking attendance for a classmate or if indeed two (or more) students have a very similar fingerprint. If the fingerprint is exactly the same, their attendance will not be checked. In this case, students (if they are in fact in class) can approach the professor by the end of the class, to prove that they are indeed in the classroom.

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). The flow of the professor's perspective is visually explained in Figure 3.1; when intended by the professor - it can be in the beginning or end of class - the attendance checking can start. Using the computer, that is connected to the projector, the professor navigates to a 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 (the link to the website will still be available on the screen, so students that did not arrive on time can still type the link). As intended by the professor, it is possible to stop the attendance taking at any time. In case the students finish the attendance checking (correctly entering the sequence of random codes) the
professor can stop the attendance, but because usually the students are running between classes, the time is manipulated by the professor, depending on how many students are in the class or still entering the room. 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, number or letters and 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 (figure 3.2). 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.

Regarding the students that, usually at the beginning of the course, have not yet enrolled, the professor can also manually mark the attendance in the website (the same happens if a student does not have a way to access the internet).

### 3.3 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.

### 3.3.1 Testing protocol

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 and numbers and letters), with 10 seconds to type each code - giving, in total, 15 combinations. The main goal was to understand, from the three different types of codes, what would be better suited for an attendance system, not being too hard but also good enough to not give too much time to send the codes to their colleagues or even take their attendance for them. 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 3.3 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 3.3: Example of a test; letters and numbers as a code type, with a 7 characters length.](image)

The different possible combinations presented to the users were done in a random order. 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*. Given the fact that the tests were done in a random order, the questions were asked to understand how the user felt about the type of code and the time to type it.
3.3.2 Results

These tests were done with students from Técnico Lisboa; these users were chosen because they are part of the target end-users of the proposed system. Testing with 25 students individually, in a class scenario - where 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, it is possible to conclude different aspects about the time to input the codes (section 3.3.2.A), the number of incorrect codes insert (section 3.3.2.B) and also the answers from the questions - Was the number of characters easy to type? and Was the time to type the code sufficient? (section 3.3.2.C).

3.3.2.A Time

Looking at figure 3.4 - a box plot with the time taken by the users to type the different code types and lengths (where blue, orange and pink represents the tests with just letters, numbers + letters and numbers, respectively) - it is possible to see that, overall, users take less time typing codes with just numbers, and take more time typing codes with letters and numbers. Looking at the same code size, for example 4 characters, the average time to input a code with just numbers is 4 seconds, whereas with number and letters is 5 seconds.

![Figure 3.4: Overview of the time taken according to the code type and length, in the first tests, where L (blue), N (orange) and LN (pink) stands for Letters, Numbers and Letters + Numbers, respectively.](image)

Using one-way ANOVA, it proven that between the code type and the time taken, there was a statistically significant difference ($F(2,1844) = 5.26, p = 0.003$). With a post hoc Tukey test, it revealed that the
significance was with the codes with Letters + Numbers, when compared with the other two codes types. Between codes with numbers and codes with letters, there was no statistically significant difference ($p = 0.76$).

Looking at the time taken for each code length (from 4 to 8 characters), analyzing separately each code type and using one-way ANOVA, it revealed that there was a statistically significant difference between the code length. For each code type, a post hoc Tukey test was made to reveal which length was the statistically significant different. For codes with letters, the statistically significant difference starts from 6 to 8 characters. With codes with just numbers, the difference was only with 7 and 8 characters. Lastly, for Letters + Numbers, based on the test, between 4 and 6 characters there is no statistically significant difference, but from 7 characters there is (with codes with less characters). Having these results, it is possible to conclude, for the three types of codes, there is a significant different when codes are higher or equal than 6 characters.

### 3.3.2.B Errors

Regarding the errors, i.e., when the users did not type the correct code (may have been because there was no sufficient time, or because some letters seemed numbers and vice-versa), the tests showed that less errors happen when the code is just numbers, and more errors when the code is numbers and letters. In the tests, 162 (8.64%) out of all the inputted codes (1875 in total) were an error; 63 out of the 162 were codes with just letters, and 72 were letters and numbers, leaving the codes with just numbers with only 27 errors. An overview of the errors, according to the code type and length, can be seen in the figure 3.5.

![Figure 3.5: Overview of the error count, according to the code type and length, in the first tests.](image)

For the errors, one-way ANOVA proved that there is no statistically significant different between the three code types with different lengths ($F(1,326) = 0.88258, p = 0.348192$).
3.3.2.C Questions

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 and numbers. Also, taking a look at the answers (figure 3.6), codes with just numbers had a significant higher score in both questions - so the students were more comfortable typing just numbers.

3.3.3 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.
I Am Here! is an Automated Attendance System (AAS), i.e., a system to track the students’ attendances and does not require much interaction from the professor. An explanation of its functionalities and how it was implemented is detailed in section 4.1 and 4.3, respectively, as well as its architecture, in section 4.2.

4.1 I Am Here!

To better understand how the system works for each type of user (professor or student), in this section it is detailed how to use I Am Here!. In its current form, I Am Here! requires all users to authenticate by using FenixEdu credentials. However, using a different OAuth-based authentication method would be trivial in a future version that wishes to implement it. Due to one of the requirements being that the system can be implemented in any course, users have simple and easy to understand functionalities. When entering I Am Here! website \(^1\), the initial page of the system is displayed (figure 4.1), so users can login.

4.1.1 FenixEdu

FenixEdu is an open-source project created in 1999, made to improve the education field \(^2\). This project helps teaching facilities to install and configure their structure - from course’s name and the professors that are teaching to the shift’s schedule and the students enrolled - in an easy way.

In I Am Here!, FenixEdu API is used as a way to login the system. Having FenixEdu also makes the professor’s job of creating a course and shift from scratch easier, since most of the relevant information

\(^1\)https://classcheck.tk
\(^2\)https://fenixedu.org/#about visited on June 27, 2019
(including enrolled students) can be directly imported from FenixEdu.

4.1.2 Professor’s perspective

The professor, as is the one that benefits and interacts more with the system, has more functionalities than the students. These functionalities that are explained in the following subsections.

4.1.2.A Create course

After logging in, a professor can see their selected courses (their name and ID), according to the current academic term (figure 4.2). These courses can either be loaded from FenixEdu or created by the user. Choosing a previous academic term, it is possible to retrieve past information - it can be used to review an attendance in a specific class, or to compare the attendance tendencies in a course, for example.

![Figure 4.2: I Am Here! course’s page.](image)

Selecting the "+" button, it is possible to create a new course in I Am Here! (figure 4.3). If FenixEdu is being used, the professors will be presented with a list of courses that, in that system, they are teaching. This means that the professor can be lecturing multiple courses (according to Fénix), but only wants to use I Am Here! in a selection of them (the professor can discard courses such as Projeto de Mestrado em Engenharia Informática e de Computadores that, although is a course, do not need attendance checking). To add one to I Am Here! it suffices to choose it from the list.

![Figure 4.3: List of the professor’s courses that are not selected.](image)

If FenixEdu is not present, or the professor wants to create a course not present therein (a professional degree or an invited lecture, for instance) this is possible by pressing "New course". It is asked the course’s name, ID and academic term. When a course is added, it will from that moment be on displayed in the list of that professor’s courses, in the landing page (figure 4.2).
4.1.2.B Create a shift

After either selecting a course from Fénix or creating a new one, the professor then has to configure it. The first step in this process is to create the different shifts it may be comprised of - in this context, a shift is a period that repeats each week, where and when a class is lectured. For instance, the same class may be taught more than once to a different subset of students, or at a different campus. So it is possible to distinguish each shift, when selecting a course, its shift’s are displayed - (figure 4.4) - the shift ID, its campus, schedule (day, start and finish time) and the room.

![Figure 4.4: I Am Here! shift’s page.](image)

The default view are the theoretical shifts, but it is also possible to choose a different type (laboratory, seminar, or other), if they exist for the selected course. Like the courses, these can either be loaded from FenixEdu or manually inserted.

At any time, the professor can add a shift, choosing the edit button. When editing the shift, it is also possible to personalize the attendance taking for any shift (figure 4.5), so every time an attendance checking is done for that course and shift, the chosen personalization is automatically implemented.

![Figure 4.5: Edit shift(s) page.](image)
4.1.2.C Insert professor(s)

Sometimes, even if a course is loaded from FenixEdu, the record of which professors are teaching it is not exactly correct. So, choosing a course, it is possible to add and/or a professor from that course. When selecting the "gear" icon, the current professors appear. It is possible to remove a professor (choosing "-" ) and/or to add a professor (choosing "+"), inserting the professor’s ID and, if the professor is not already in the name, the full name (otherwise, the name can be omitted).

![Figure 4.6: Add/remove a professor functionality.](image)

4.1.2.D Associate a professor to a shift

While editing a shift, it is possible to associate a professor to a shift (or vice-versa). Associating a professor to a shift, means that a shift is lectured by that professor - for example, in a faculty with two campus, usually, there is one professor responsible for each campus’ shifts. This does not mean another professor can not substitute the main professor of a shift (for example, when the main professor is sick or could not make it to class), but when a professor selects a shift that is not associated, an alert pops up "Attention: this is not your usual shift". With this, the possibility or error (i.e., when a professor selects, by mistake, a shift that is not his/hers) decreases.

4.1.2.E Insert/Update student(s)

In I Am Here!, a user can either be a professor or a student - a professor can be a student but the opposite is not true, meaning that a professor can take attendances but a student does not have access to the professor’s dashboard. As explained in the previous section 4.1.2.C, it is possible to manually insert a professor (the other options is when the professor logins and, according to FenixEdu, if they are a professor, they are labeled as that). The students are the ones that are not labeled as professors. These are also subdivided in two categories, depending on the course: enrolled or not. Being enrolled or not can come from FenixEdu, or it can be manually inserted by the professor.
In figure 4.2, selecting the bottom arrow, it is possible to import students (figure 4.7). This functionality is useful when the professor wants to change the student's name and/or add the student's number. Using FenixEdu, in some cases, the student's Fénix id (e.g. ist142083) is different from the student's number (e.g. 82083). So, if the professor needs both numbers, Fénix only gives the id (and not the student number). Any professor has access to the student Fénix id, number and name. Even using a login different than FenixEdu, having this, the professor has both student identifications and can choose which one to use. The same happens with changing the name of the student; it can also be helpful, in case the professor only wants the name and the surname. Choosing the Course, the student is automatically enrolled in that course. The line format is the following, where each students is in a different line:

ist182083,68083,John Smith
ist182084,68084,Caroline Altman
(...)

Figure 4.7: Import students functionality.

After inserting the students, selecting "Import", students are imported. Otherwise, selecting the bottom arrow (that now is in blue) or the "Cancel" button, the importing in canceled. Either way, the user is redirected to course's view (figure 4.2). It is also possible to load enrolled students from a specific course from FenixEdu (it is explained in the following section 4.1.2.F).

4.1.2.F Start a new attendance

Having selected a course and shift, it is possible to check previous classes (figure 4.8). By default, the user's classes are displayed, but, selecting a different professor, it is possible to check the other professors' classes as well.

Pressing the "+" button, an attendance checking personalization begins. To begin the attendance checking, first, it is required to personalize it, defining the different aspects, as it is possible to see in
The default settings are the ones retrieved from the tests made in Understanding Code Entry (section 3.3) or, in case it was edited by the professor, the one that was personalized when editing or adding the course (mentioned in figure 4.5):

- **Number**: the number displayed is always one number higher than the previous class.
- **Title**: is optional, it can be used to detailed the subject lectured in that class.
- **Extra Class**: defines if the class is a normal one, or an extra class; an extra class can be an invited, doubt or other lecture.
- **Type**: defines the type of code that the students will have to input: letters, number or letters and numbers.
- **Length**: defines the length of the code, in a range of 3 to 8 characters.
- **Time for each code**: defines the time to input each code, in a range of 8 to 12 seconds.
- **Consecutive codes**: defines the number of consecutive codes required to take the attendance correctly, in a range of 2 to 6 codes.
- **Access**: defines if the attendance only accepts enrolled students (or not).

When the attendance session is created, the QR Code and link appear (figure 4.10), so students can access the attendance link. When intended, the professor can start the attendance.

Selecting "Start", the attendance starts; a code is display on the screen, as well as the time left to input it (figure 4.11). On the left, the QR Code and link are still available, for the students that did not managed to get it before. On the right, the student’s id appear according to the last one that checked the attendance.

The countdown gradually fades from black do red, as a gradient, to give a visual feel of the time left (so the students do not have to look for the seconds that are left, and only the countdown bar). When the time of the code is 0 seconds, it generates a new one, until the professor stops the system.

The professor can “Pause” the attendance, and it goes back to the previous view (with attendance links, figure 4.10, where it is possible to "Start" (again) the attendance, or can "End session", and it redirects to that class.

### 4.1.2.G Insert a previous attendance

Another type of attendance can also be added (selecting the **bottom arrow** button, on the top right of figure 4.11), which is when a class occurred and the system was not used - if there was an energy
**Figure 4.8:** *I Am Here!* attendance’s page.
failure, for example - the attendances can be imported. It requires the professor's id, the shift, class number, title (that is optional), and the student's id and their type of attendance (late or not):

On time: ist182083, ok
Late: ist182083, late (late)

4.1.2.H Attendance history

When an attendance is finished, the professor is redirected to their general attendance history (figure 4.8). In the table, each line represents a different class, ordered by date, from the first to the last class. This table has the general information of the attendance: class number, title (this field can be empty), date, code type (L, N or LN, as letters, number or letters and numbers, respectively) and length (in number of characters), number of consecutive codes required and the number of students that attended. If a class was created by mistake, it is possible to remove it. In case the professor wants to sort the table by the number of students that attended, or other column, it is possible, selecting the table header intended to sort by.

Choosing a specific class, it is possible to analyze the students that attended it. The students that took the attendance appear on the left side of the screen, as well as the students that tried but did not managed to accomplish the attendance, on the right side table (did not manage to insert the correct number of consecutive codes or took to long to type the codes, for example), as it is shown in figure 4.12. The table on the right is useful to quickly add the students in those cases, but, if the student did not arrive on time for the attendance checking, but still was in the class, it is possible to manually add the students - in the bottom of the left table, it is an input, where it is possible to insert the student's id - and to specify it is a late (or not) attendance. Looking at the column “On time”, the students with a green check attended on time, and the red cross are the late ones. At any time, the professor can change the
student's status from *late* to *on time*, and vice-versa, by just selecting on the column "On time" on the specific student's row.

4.1.2.I  **Edit class information**

It is possible to edit the class name, number and if it is extra (or not). During the editing, it is possible to cancel the action.

4.1.2.J  **Export attendances**

Selecting the *link* button in a course, shift or class, all attendances from that selection can be exported in a *TSV* (Tab Separated Values) format. After pressing the button, a *prompt* is displayed in the center of the screen, where it is possible to copy the link. The information is exported as follows:

Professor number, Student ID, Student number, Student name, Attendance type (late or not), Class number, Shift

The name and student number can be the ones imported by the professor (as it is explained in the previous section 4.1.2.E; if the professor has updated the students name and number, otherwise the name will be was FenixEdu gives (the student's full name) and the *Student number* will be empty.

4.1.2.K  **Detecting fraudulent attendances**

In figure 4.12, two students that have their row highlighted in yellow, because the system found their device's fingerprint too similar to not be the same. This gives the professor the visual information that
those students may be trying to cheat the system. In this case, or anytime, it is possible to remove a student from an attendance, pressing "-" in the respective row.

### 4.1.2.L Student’s information

Selecting any student, a brief summary of the student’s attendance history is displayed in a table (figure 4.13), where it is possible to check which classes the student has attended, the shift and campus of the class, and the student’s IP address and User Agent, in columns **Class #**, **Shift**, **Campus**, **IP** and **User Agent**, respectively. In this section, the professor can confirm and/or check the attendance history of a specific student.

### 4.1.2.M Course’s attendance history

After choosing a course, selecting **Students** - the tab, on the left corner, next to **I Am Here!**, all enrolled students appear in a table, with the student’s Fénix id, name and the number of classes that each student has attended. This table can be exported in a TSV file.

### 4.1.3 Student’s perspective

This perspective is simple so students can take their attendance in a quick way, using any type of device (it only requires to have a browser and network access).

In figure 4.10, at that moment during class, students can scan the QR-Code or type the link to go to the attendance link, depending on the device they are using. If the link is invalid, i.e., the link does not correspond to the one that is on the professor’s screen, an error message shows up (Invalid...
Figure 4.12: Class view, with the students that attended the class and also the ones that tried. The ones highlighted in yellow, according to the system, have a very similar fingerprint.

Figure 4.13: Brief summary of a student's fingerprint.
When entering the correct attendance link, according to the specifications personalized by the professors, the students have to accomplish them; an example scenario can be: enter 3 consecutive correct codes, having 10 seconds to type a 6 characters code. Another way of entering the attendance link is by typing the ID of the link.

Depending on the type of code (letters, numbers, letters and numbers), the student’s keyboard differs. If it is just numbers, the number’s keyboard appears on a mobile version (on desktop, it only accepts numbers). Otherwise, the normal QWERTY keyboard is displayed and numbers and/or letters are accepted. During the insertion of codes, when entering each code, the student receives an information if the codes is correct (Correct code.) or not (Invalid code.) - figures 4.14 and 4.15, respectively.

When the task is complete, the students receive a message (Correct! Attendance check!), and the input disappears (the student no longer needs to type codes). As explained in the previous subsection, the student can also check that their attendance was checked by looking at their number on the right side of the professor’s screen.

### 4.2 Architecture

The system consists of two major components (figure 4.16): the front-end and back-end. The back-end is subdivided in two components - the server and database. All the information is stored in the database. The server directly interacts both with the database and the front-end. It processes all the information and provides specialized functions to retrieve the information from the database that is required for 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). Regarding the sources of information needed to develop the system, to login, it was used the FenixEdu Open API, and, for the front-end, some resources were used to improve the system. How it was implemented and why, is explained in the following section 4.3 - Implementation.
4.3 Implementation

As soon as the general idea of the desired attendance system was reached, the implementing began in a prototype, as mentioned in 3.3 - Understanding Code Entry, which was used thereafter to experiment with and validate the solution. I Am Here! was implemented using JavaScript, HTML5, CSS and, for the database, MySQL.

After researching different types of languages, JavaScript 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, JavaScript 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 (such as PHP, for example). This is useful to have a timer that constantly change the system’s status - which, in this case, it is to update the attendance session every second. 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. Having JavaScript in both front and back-end is good, so it is possible to focus in only one language. For the database, having already experience with MySQL, it was the software chosen to use as the database. To work with MySQL, MySQL Workbench was used as a visual interface to manage the database.

Through the implementation, some of the proposal of the system was done, and others were discarded or proved to not suit the system. What was done and what was not is explained in the following sub-sections.

4.3.1 Server

Throughout the testing phase, I Am Here! was (and still is) running in a virtual machine hosted at Rede das "Novas" Licenciaturas (RNL). Before this, Sigma’s Cluster was the first option, but because it can not maintain a JavaScript system running for more than a few hours - and the intended solution was to have the system always available, at any time, without having to run it by hand - this option was discarded.
Having the system running in JavaScript, by nature, it is already asynchronous, so the system that is running does not have to block or wait for a response. Having this, it is possible to have more than one request at the same time and, if one of the requests takes longer, the others can be done and completed, without having to wait. This type of system is more suitable for real world scenarios due to the fact that it does not have assumptions of order or time of the events.

After creating the server, displaying a simple index.html page, HTTPS (Hyper Text Transfer Protocol Secure) was the first implementation made in the system. To implement this, first, a domain was chosen for the website - classcheck.tk. Because this link could probably be written in a computer or displayed in a screen multiple times, it was chosen a short, but related with the scope of the system, link, via Freenom (Freedom Registry) \(^3\), as it is the operator that runs the (free) .tk domain (as well as others). Then, to generate the HTTPS certificate, it was used Let's Encrypt \(^4\), as it is a free provider, having the only catch that the certificate only lasts three months, so it is required (and recommended) to generate one, at least, every two and a half months (a step-by-step guide on how to update HTTPS certificates is in appendix F). The communication between front-end and server is encrypted and can not suffer man-in-the-middle attacks, i.e., because of HTTPS, the communication between two ends can not be understood by a third and not authorized party.

Catching on after the prototype mentioned in section 3.3, the system then began to be imagined with all of its components, as displayed in figure 4.17. The server.js only communicates with Service.js, so it is more organized and only has one intermediate point of communication. The default config.js and config.js are the configurations needed to run the system (it has the database's password, the HTTPS' certificates, and many other important information); the default one is public (on the GitHub) and the other one is not, since it has the way to access to all the information in the database. The Service.js is the one responsible to communicate with the remaining parties - Code.js, fenix-api.js and Database.js. In Database.js is where all the queries and procedures are made or called for the database. For the fenix-api.js, all the request are made for the FenixEdu API are made here. In Code.js is where all the properties and functionalities of the attendance are made. In the server itself (server.js) is where all requests from the front-end are handled. This requests can either be a POST or a GET.

### 4.3.1.A Authentication

Having the system secured, FenixEdu API was choosen to use to login in the system. The implementation was done following the steps of FenixEdu official page \(^5\). Since the system was firstly implemented to be used in a Técnico Lisboa's course, using this API made the part of retrieving important information

---

\(^3\)https://my.freenom.com visited on June 11, 2019
\(^4\)https://sslforfree.com visited on June 11, 2019
\(^5\)https://fenixedu.org/dev/tutorials/use-fenixedu-api-in-your-application visited on June 11, 2019
Besides the login part, the system is not dependent on FenixEdu - it is possible to create and personalize all the information needed to take the attendances, from its courses to shifts and students. Since *I Am Here!* uses OAuth \(^6\) (platform that grants access, in this case login, without having to deal with password management), it is very simple to change the authentication to other company (such as Google, Twitter of Facebook), depending on the field where it is going to be implemented.

In the beginning of the system implementation, when the professors entered *I Am Here!*’s website, the server crashed (and restarted). This kept happening, giving the error Can’t set headers after they are sent to the client. After searching about the problem, it was discovered that, since JavaScript is naturally asynchronous, and because, after logging in, if the user is a professor, it does more requests and some of the requested were ending and others had not started yet. So, when the professor logged in, the server kept getting an 502 Bad Gateway error. The solution was to only end() the functions inside the callback function in the server.

### 4.3.1.B FenixEdu API

All the information from FenixEdu API is requested with a GET method. Each request has its endpoint and the information received is then handled by the services, so it ends up where it is wanted (it can be stored in the database and/or displayed in the front-end).

From this API, besides the login part, the requested made were:

- **/PERSON**: to get the user’s role (student and/or professor). If the user is a professor, it is possible to collect the course’s name and id.

- **/COURSES/ID**: to get the course’s academic term

- **/COURSES/ID/STUDENTS**: to get the enrolled student in a specific course

---

\(^6\)https://oauth.net/ last visited on July 4, 2019
For all requests, ID is the unique identifier of each course, that comes from Fénix.

4.3.1.C Services

The services are the intermediate point between the server and the remaining parties, it handles all the information that crosses the back-end. It provides the required services.

Depending on the request, services functions' either call the database, make a request to the FenixEdu API or handle the attendance workflow. These functions are then called in the server. This keeps the code organized and clean (easier to manage).

In the services is also where all the information is arranged. For example, when requesting the student’s that attended a certain class, the services handle and organize it, to be sent to the server and then displayed in the front-end, informing which students are late and also those that the system found a very similar fingerprint. It is the services job to stage all the information that comes across.

4.3.1.D Attendance Workflow

It is possible to have multiple attendances running at the same time (from a different professor, or not). To assure that these do not collide with each other, it is required to have some properties for each attendance.

The attribute that differs an attendance from another is the randomID, that is unique. The attendance’s personalization (code type, length and number of seconds to type each code) is stored with the randomID (so it is possible to see which type of personalization was done in each class). Each code generated is numbered, starting on 1, and so on. This is used to verify if a student has inserted the correct number of consecutive codes. Each time a professor stops or closes an attendance, the attendance link becomes invalid (i.e., the attendance is not running). With this, the attendance link becomes invalid (since it is not necessary to use it). Two very important attributes are time\_ms and the INTERVAL. These are correlated. The first one sets how long the code is being displayed, and the other one is the number of seconds the professor has personalized to have the same code running, i.e., the time each student has to enter the code once it is displayed for the first time - once the time\_ms is equal to INTERVAL, a new code is generated. If the professor personalized the attendance so only enrolled students can take it, a list of the enrolled students is request, so only they can access the attendance link and insert codes.

A – Synchronization To assure the system is on track, every second the server updates the front-end with the time left (time\_ms) to insert the code. According to the time left, the code inserted by the student is compared with the current code (i.e., the code that is being displayed on the professor’s screen), to validate if the code is correct or not (and, thereafter, if the students have checked their attendance). This means that the code is available for the exact time personalized by the professor.
However, in practise, it is available according to the latency - time taken to make the request and get a response; the larger it is, less time is available, as it is not taken into account by the server. Meaning that, if a professor sets 10 seconds for the time to input each code, in reality, it is lower, because the latency is not zero - if the latency is 1 second, students have around 8 seconds to successfully input the code, that is minus 1 second for the code to be shown on the projector and minus 1 second for the code to be sent by the student. Giving the scenario where the system was tested, the latency was around 100 milliseconds, value that is to low to be considered a problem (or even being noticed).

### 4.3.2 Database

The system stores all the information in the database. In figure 4.18, it is possible to see I Am Here!'s database schema, with its tables and columns.

To avoid SQL Injection attacks, i.e. when there is a vulnerability in a input that is used directly in an SQL query, all data provided from the user is escaped.

```javascript
1 database.prototype.getShiftsByCourseID = function (courseID, callback) {
2     var sql = "SELECT * from Shift WHERE courseID = ?";
3     var arguments = [courseID];
4
5     this.pool.query(sql, arguments, function (err, rows, fields) {
6         (...)
7     });
8 }
```

In line 2, each ? is replaced by an escaped version of each of the arguments (line 3). For example, boolean variables are converted to true/false. Having pool.query() (line 5) it is possible to have multiple calls running at the same time.

To access all the information, a set of queries, stored procedures and a function were created in MySQL. To access simple information (e.g. Select the courses from a specific professor), a query was made to the database. If a specific query is complex and/or is executed repeatedly (e.g. Select all the attendance information from a specific shift, course and professor, ordered by date), it is commonly transformed to a stored procedure; so, instead of constantly calling the same query, it is already stored in the database and it is executed more efficiently.

In this system, in the database, only one function was implemented, due to the complexity of the query, as it is used to check if a student has already checked his/her attendance. It checks the code

---

7https://acunetix.com/websitesecurity/sql-injection visited on June 15, 2019
8https://github.com/mysqljs/mysql#escaping-query-values visited on June 19, 2019
sequence and, depending on the number of consecutive codes required, it inserts (or not) a student to an attendance.

4.3.3 Scalability

Professors can do many things with the system, but the one where the server needs to handle more requests is during the attendance checking. It begins when the student need to access the attendance link. Then, when the attendance taking starts, each second the server handles request. From the professor's side, the current time left, code and the students' ID are updated. From the students' side, each gap to insert a code (specially in the first few codes), the server is overloaded with the codes inserted; also, as the students need to have a response (if the code is correct or not), the server also responses with that after each code inserted. In this scenario, the system has handled over 100 students during an attendance checking (a total of 106 students), and it performed as expected. This is one of the many perks of using JavaScript. Being asynchronous, it is possible to have multiple nodes running at the same time, each one doing its task, without having to wait to start and/or finish.\footnote{https://quora.com/How-does-asynchronous-IO-improve-scalability visited on July 6, 2019}

When the system was proposed, we thought that it would be helpful to tell the students how many codes are left to complete the attendance taking. After a few weeks of the system's implementation, this option was discarded because of the number of requests that are already made during the attendance (and adding the number of codes left would overload even more the system).

4.3.4 Setbacks and Improvement

To have a simple attendance system implemented for the first class of the second semester, besides the attendance part, the system only had an idea of class. During the first use, with around 60 students in class, the system began working fine and then, after a few minutes, it failed (and restarted). At this point, a problem was discovered. At each second, the server requests for the instance of Code, i.e., all the attendance information (current code, time left, etc). Because the system restart, that instance of Code was "removed", i.e., it was gone when the server went down and it was not initialized again. After figuring the problem, the solution was to, when the server goes down, it requests the database the attendance information and initializes again that instance of Code (instead of just requesting the Code, that could not be initialized).

Throughout the implementation of the system, it was building up to be more robust and complete. After the first weeks of using the system, having the attendance part working well and with no problems, the system began to extend - adding the idea of course, shift, academic term and a few other needed functionalities. These could either be loaded from FenixEdu or manually inserted by the professor.
Figure 4.18: *I Am Here!* database schema.
Adding the idea of course, the first implementation was to request the professor’s courses (given by the FenixEdu API). With this, it was possible to retrieve the name and ID of the course, as well as the shifts. But, in case a professor wants to check the attendance in a situation outside a specific course - for example, in a conference, or an invited lecture - it was implemented the option to create a course from scratch, with the same structure of the Fénix’s courses.

Because the course where the system was implemented had a bonus for the attendances, it was required that the attendances from I Am Here! could be exported, to be displayed in another platform. With this, it was implemented the option to export in a TSV (Tab Separated Values/Variables) file the attendances; it could be the attendance from a specific class, shift or all the attendances from a given course.

4.3.5 Front-end

As explained, the front-end was implemented using just JavaScript, HTML5 and CSS. Due to the simplicity of the system, not many resources were used. Having less requests, it is possible to serve more users at the same time. Throughout the system implementation, it began growing to be more sophisticated.

There are five resources used in front-end - fingerprint2.js, qrcode.js, W3.CSS, W3.CSS Pro and W3.JS. The first one, fingerprint2.js, is implemented in the attendance link that the students access, and it is used to retrieve information from the student’s device. The second one, qrcode.js, it is used to generate the QR Code that is displayed on the professor’s screen. The other three, W3.CSS, W3.CSS Pro and W3.JS, were used to do simple improvements in the system - from having animations in the buttons to sorting tables. With this five resources, it was possible to implement simple features without having to reinvent the wheel.

Because each user has its different functionalities, the professors’ and students’ side have its different html pages. Each section of the professor’s side (explained in 4.1.2) has its different html file. In figure 4.19 it is only represented (a part of) the professor’s and student’s file organization, to get an idea of it.

Both professors’ and students’ side make POST and GET requests to the server. A POST is requested when the exchanged information alters the back-end information (e.g. Insert students ist182083 to the database.), unlike GET requests, that are used when the exchanged information retrieves details from the database (e.g. Get all the students that attended class number 4.). The information itself is exchanged through JSON (JavaScript Object Notation), as it is a language-independent data-interchange and easy to understand format. An example of a GET request both professors and students do is their name,
that appears on the top right corner of the screen.

All the icons used are Unicode characters, as it does not overload the system with more request, as opposed to Favicon and many others sources of icons or images.

4.3.5.A Professors’ side

In the professors’ perspective, there is a couple of tables and selects; all these were done using \texttt{<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. All the information displayed in these are \texttt{GET} requests made to the server.

Both “Import Students” and “Import Attendance” have a \texttt{<textarea>}. 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 \footnote{\url{https://regex101.com} visited on June 23, 2019}.

The sorting functionality implemented in the tables is from \url{W3.JS} resource. It is using the \texttt{HTML.sort()} function \footnote{\url{https://w3schools.com/w3js/w3js_sort.asp} visited on June 23, 2019}, that receives the table’s id and the column that it wants to sort.

The countdown bar was made using only JavaScript and CSS. It is a \texttt{<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.

A – Responsive Design

It is expected that, from the professor’s perspective, they 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.
4.3.5.B Students’ side

The student’s side only has two HTML pages: one for the login and other, to input the codes. Students do not have much information to see when they enter the attendance link, only their full name and an input, to insert the codes. It is not possible to insert a null (empty) code. Every time a code is inserted, a message informing if the code is correct or not is displayed. Depending on the response, it can be presented in red, if the code is not correct, or otherwise, in green.

In the beginning of the prototype, 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:

```html
<input maxlength="15" type="text" id="inputstudent" style="text-transform: uppercase" autocomplete="nope">
```

The input only accepts a code with maximum 15 characters (no need for more, because the codes can only have a maximum of 12 characters), so students can not insert random codes that could potentially overload the system and occupy space in the database for no reason. The default type of the input is text, but, as explained in the previous section 4.1.3, if the attendance only has codes made by number, the type changes to number.

A – Responsive Design  
Opposed to the professor’s perspective, with the students, they 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>` to use the method viewport. 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. For example, having:

```html
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

The `width` is used to fit the page according to the screen-width of the device (this is very helpful mainly for mobile users).

---

18 https://what.thedailywtf.com/topic/20062/seriously-browsers-that-s-how-autocomplete-is-supposed-to-work visited on June 11, 2019
19 https://w3schools.com/tags/tag_meta.asp visited on July 4, 2019
20 https://w3schools.com/css/css_rwd_viewport.asp visited on June 28, 2019
4.3.5.C Device Fingerprint information

When a student enters the attendance link, it immediately retrieves the User Agent and IP address from the HTTP headers attributes and scans the student’s device fingerprint, using fingerprint2.js. This resource can extract many different aspects of a device’s fingerprint - from its plugins to the canvas. To have enough data to evaluate the student’s device, all the information available in this resource was collected. The information is sent in a JSON as a POST to the server, to be stored in the database (according to the current class).

In the context of I Am Here!, the most common fraudulent attendances are:

- Case 1: Sending the codes via SMS or other type of chat to a colleague;
- Case 2: Video chatting with a colleague, showing the codes that are being displayed;
- Case 3: One student taking an attendance for a colleague.

The first case is hard to achieve. A student has to insert the code (for himself) and also send to a colleague. From the study made in 3.3, and giving the short amount of time a student has to accomplish this, this case is unlikely to succeed - by the time the student has inserted a code, even if they can send it, the colleague that is outside does not have time to insert code during the right time.

For case 2, I Am Here! does not have a way to directly stop from happening. Indirectly, the video call can make it difficult to read the codes of the projector, depending on the device’s and application capability, and the network conditions. The professor can also notice and that puts the student’s attendance on the line.

For the third and final case is where the device fingerprinting works. From the information that is retrieved from the students’ device’s, its importance (i.e., the level of importance and if the information differs if the user changes the browser) can be checked in the fingerprint2.js official page.21 If a student uses the same device (independent from which browser it uses) to take two (or more) attendances, the IP Address stays the same, so that is the attribute that is going to uncover this type of fraud cases (i.e., it is the most unique and important attribute from a device’s fingerprint). Since each fingerprint is stored for each class, if a student uses a different device (it can be a new purchase device or, in a worst scenario, it can be a student doing it for a colleague), the remaining attributes (browser independent and dependent) are going to be taken into account and those cases will also be flagged, because it will be different from the previous classes (and usually, students do not change computer/cell phone regularly).

5 Evaluation

Contents

5.1 Classroom Usage .................................................. 55
5.2 User Evaluation .................................................. 62
5.3 Device Fingerprinting Evaluation .............................. 67
5.4 Discussion ....................................................... 68
To cover all its aspects, *I Am Here!* was evaluated in 3 separate ways: a Classroom Usage Evaluation, an User Evaluation and a Device Fingerprinting Evaluation. 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. For the Classroom Usage Evaluation, 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 (5.1.1) and Efficiency and Error Rate (5.1.2). For the User Evaluation, a group of users were asked to complete a set of tasks, meant to cover most of the professor’s functionalities. Lastly, the Device Fingerprint evaluation was done to evaluate the system’s effectiveness in catching fraud attendances. Each metric was evaluated and discussed. In the end of this section, an overall discussion of the system’s evaluation is done, in section 5.4.

5.1 Classroom Usage

*I Am Here!* was implemented in *Multimedia Content Production* (MCP) course, during the second semester of the school year 2018/2019. In this academic term, classes started on February 18, 2019, and ended on May 31, 2019. *I Am Here!* was used in 23 classes in both Técnico Lisboa’s campus - Alameda and Taguspark - giving a total of 46 classes. In this course, 116 students were enrolled and using the system. For this evaluation, it is only considered the results from Alameda’s classes and it is important to know that classes were on Mondays and Wednesdays, the attendance taking was done in the beginning of the class and also there were three badges (a bonus) for attending 50%, 75% and 100% of the classes, respectively. In this course, there was an evaluation every Monday, and there was one (from the 23 classes) extra class - where Masters’ Theme Proposals were presented, as it is possible to check in figure 5.1. In blue marks the evaluations (called quizzes), in green the extra class (the Masters’ Theme Proposals) and in purple (represented by H), the holidays - in March (2 days for Carnival), April (1 week Easter break) and May (Portuguese holiday) - that overlapped MCP’s classes (this overlap caused a break of at least one week without using the system).

<table>
<thead>
<tr>
<th>Date</th>
<th>Feb</th>
<th>Mar</th>
<th>Apr</th>
<th>May</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class number</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>20</td>
<td>25</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>11</td>
<td>13</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>H</td>
<td>22</td>
<td>24</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>17</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td></td>
<td>H</td>
<td>8</td>
<td>13</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>27</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 5.1:** Multimedia Content Production time line, in Alameda’s shift, through the second semester the school year 2018/2019.

The system’s Effectiveness (5.1.1) and Efficiency and Error Rate (5.1.2) are described and evaluated in the following subsections. The resulting data from this evaluation is in appendix A.

55
5.1.1 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 tacking, the number of students was manually counted (the students that arrive late will not be taken into account in this metric). Because the counting was not supposed to be obvious for the students, and due to having students entering/leaving the room, the number may differ by 1 or 2 students comparing with the real number of students that attended each class. This slight different does not have a significant impact in the calculations. To calculate this metric, it was used its respectively formula:

\[ \frac{N_{\text{registered}}}{N_{\text{present}}} \times 100 \]

Where \( N_{\text{registered}} \) is the number of attendances registered by the system, and \( N_{\text{present}} \) is the number of students in class (which were counted by the author).

In figure 5.2, the number of students physically in class (in red) and the number registered by the system (in blue) is represented for each class. As it shows in figure 5.3, 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 or, because of this bonus, it could be that the number of attendances left was lower than the number of attendances needed to achieve the next badge (so students quit taking their attendance, because it was not necessary anymore). By the end of the system’s implementation in the course, due to the reasons mentioned before, it had an average of 95.05% of effectiveness.

![Number of students: in class vs registered by the system](image)

**Figure 5.2:** Number of students in class: physically and the ones registered by the system.
Figure 5.3: Difference: students in class and the ones registered by the system.

In classes 17 and 18, the effectiveness was higher than 100% because the number of students counted was less than the number of students registered by the system. Because the number counted may differ by 1 or 2 from the real number, these cases might be affected by that. Also, due to the fact that some students did not check their attendance during class, the number of students physically present in the system ended up being bigger than the number registered by the system, giving an average effectiveness lower than 100%. Besides this, students that were inside the classroom and wanted to check their attendance, did it on the right time (except in class 20, where the internet did not cooperate).

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

\[
\frac{N_{expected}}{N_{real}}
\]

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. It is expected that in the beginning of classes the students should have a high error rate and, gradually, this should be close to zero. The opposite is expected for the efficiency: in the beginning, it is expected to be low and, gradually, as the students get used to the system, will be higher.
The average of inserted codes for each class is visually displayed in figure 5.4 - the dark lines marks the classes with evaluation, and yellow the ones where the students were a period of time without using the system. The system ended up with an average efficiency of 82.80% (figure 5.5) - the efficiency was always above 65%, and the trend line shows that the system was gradually getting a higher efficiency, which means students were getting more used to the system from class to class. In figure 5.9, 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.

Figure 5.4: 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).

It is also important to take into account the duration of the attendance checking. Figure 5.6 displays the time, in minutes, the system was available, so students could enter the codes. The system was running for the longest time in the second class. In this class, students had doubts about the course's evaluation, and the professor kept the system running while answering questions (having the system running during almost 9 minutes). After this class, as the students were getting used to the system, the attendance checking ended up being running and available for around 2 minutes. Since each code was available for 10 seconds, in these classes, the system displayed a total of 12 codes, giving a significant gap for students to enter the number of required consecutive codes.

Looking at figure 5.4, 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
attending. As of the fifteenth class, the average number of inserted codes dropped to less than 4, given that students were getting more and more used to the system.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{efficiency.png}
\caption{Efficiency (in \%) for each class. The darker blue marks the efficiency in each class, and the light blue the tendency line.}
\end{figure}

\begin{figure}
\centering
\includegraphics[width=\textwidth]{time.png}
\caption{Time (in minutes) the attendance system was running for each class.}
\end{figure}

In figure 5.7, a panorama of the number of codes inserted in each class is displayed. It is possible to see that the majority (shown in blue) of the students stick between 3 to 4 codes inserted. The dots (for example, in class 9 or 22) or the loose ends (in class 5 or 21), are the cases where a student was not consistently going to class, and that justifies the high number of codes inserted - for example, in class 5 (the blue loose end) and in class 9 (the blue dot) inserted more than 15 codes - meaning that these students had recurring problems using the platform and therefore did not become comfortable using it. In the first 4 classes, it is possible to check those who did not insert the minimum number required (3).

Taking a closer look to the number of inserted codes for each student, it is possible to check when each student became comfortable using the system, meaning, when did they start inserting the correct
Figure 5.7: Number of codes insert in each class, where each line represents a different student. Grey represents students that were constant, the remaining are those with recurring problems using the platform.

number of codes required. In figure 5.8, it is possible to see that from the first three classes, a total of 31 students (out of 72) were already inserting the correct number of codes, meaning that became comfortable using the system. Minus 4 students that never learned how to use the system - either because they quit the course or did not go to enough classes to get used to (represented in figure 5.8 by . . .) - nearly half of the students managed to learn the system in the first three classes. The remaining students, throughout the classes, did eventually get used to using the platform.

Figure 5.8: Number of students that became comfortable using the system in each class.

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 5.9 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.

![Error count: number of invalid codes per class](image)

---

**Figure 5.9:** Error rate: number of incorrect codes inserted per class.

On the seventeenth class, the countdown bar 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).

For the error rate metric, it was expected that the number went evenly from high to low, but it was not the case. The explanation can be what was mentioned: the codes could be hard to memorize, the number of students in class was not always the same and also the skipped classes. For the efficiency metric, although the system was not being used constantly by the same exact students (since some students only went to half of the classes), the efficiency ended up with an average efficiency of 82.80%, meaning that, even with the errors made by those students, overall the attendance was taken efficiently (mostly all students did take their attendance, and it took a little amount of time).
5.2 User Evaluation

With a set of tasks, I Am Here! was submitted to an User Evaluation, in order to understand how it is perceived by users. The tasks evaluated are described and discussed according to the achieved results. All questionnaires and data from this evaluation are in appendix B, C and D.

5.2.1 Method

To validate the solution, 20 users were asked to perform 7 tasks, meant to cover most of the functionalities provided by the system. The chosen users must be comfortable around computers. Before the tasks, a brief explanation of the system was done, detailing the scope of the system. Also, the users had 5 minutes using and getting to know the system before starting the tasks.

The tasks were:

- **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.

Neither one of the tasks is dependent on another and were all done using a computer. For each user, the order of the tasks was done in a random order and the beginning point (the first page of I Am Here!’s website) is the same for every task. Since users had 5 minutes to get to know the system, some tasks should be easy and straightforward. Those tasks are: 1 (Create an attendance session), 2 (Check if a student attended a specific class), 3 (Get the permanent link) and 6 (Create a course). These tasks are easy because the way to do them it in the center of the screen (i.e., where the user is constantly looking).
Also, since the functionality is also represented in the correspondent button (for example, task 3 - Get the permanent link - its button has a link character to represent it), it eases the job of completing the task. The remaining tasks (4, 5 and 7), since those require several steps which may not be immediately obvious after a cursory inspection of the interface, are labeled as hard tasks.

In each task, the time to complete the task, number of errors, number of tasks completed correctly (and incorrectly) was collected. Since there is only one way to go to each functionality, the number of clicks will be proportional to the number of errors, so it will only be taken into account the number of errors. For each tasks, a 5 minute limit was given to complete it. All the comments and suggestions given by the users were also noted.

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) was asked (both questionnaires are in appendix C and B, respectively).

5.2.2 Results

Users that tested the system averaged 27 years old (8 women and 12 men). All of them use a computer on a daily basis and only one of them was a professor. Out of the 20, 8 were also participants of Understanding Code Entry (section 3.3) tests. Neither one of the 20 were enrolled in MCP course or connected to the system whatsoever. All tasks were completed correctly. The remaining results are discussed separately, according to each metric evaluated.

5.2.2.A Time to Complete Task

Having experience with I Am Here!, the tasks could be done quickly and all take around 20-30 seconds to complete. Figure 5.10 shows a box plot with the time required to complete each task and table 5.1 shows computed statistics: median, average and standard deviation for the time to complete each task (complete results can be found in appendix E).

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 5.10). Also, the standard deviation in these tasks was also higher (as per table 5.1), compared with the remaining tasks.
Figure 5.10: Time to complete each task.

<table>
<thead>
<tr>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Median</td>
<td>52.00</td>
<td>48.00</td>
<td>22.00</td>
<td>51.00</td>
<td>90.50</td>
<td>36.00</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>52.50</td>
<td>48.05</td>
<td>21.90</td>
<td>60.50</td>
<td>105.30</td>
<td>36.65</td>
</tr>
<tr>
<td></td>
<td>Standard Deviation</td>
<td>9.54</td>
<td>8.76</td>
<td>7.63</td>
<td>40.14</td>
<td>51.44</td>
<td>7.97</td>
</tr>
</tbody>
</table>

Table 5.1: Time (in seconds) to complete statistics for each task.

5.2.2.B Errors During Task

Figure 5.11 shows a box plot with the resulting errors in each task and table 5.2 shows computed statistics: median, average and standard deviation for the number of errors in each task.

<table>
<thead>
<tr>
<th>Task</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Median</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td>4.50</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>0.10</td>
<td>0.05</td>
<td>0.20</td>
<td>1.05</td>
<td>4.50</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td>Standard Deviation</td>
<td>0.45</td>
<td>0.22</td>
<td>0.70</td>
<td>1.23</td>
<td>1.32</td>
<td>0.64</td>
</tr>
</tbody>
</table>

Table 5.2: Errors statistics for each task.

It is noticeable that the tasks that took more time (tasks 4, 5 and 7) were also the ones that had higher errors by the users. In this case, the standard deviation is not large, since all users had a very similar (high) number of errors. Every time a user selected an incorrect button, an error was counted. As predicted, these tasks took more time because, according to the users, the way to complete/achieve the menu was not obvious. 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 5.11: Errors during each task.](image)

### 5.2.3 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 5.12 shows a box plot with the resulting answers, from 1 (Very Hard) to 7 (Very Easy).

![Figure 5.12: SEQ (Single Ease Question) answers for each task, where users had to answer in a scale of 1 (Very Hard) to 7 (Very Easy).](image)

From figure 5.12 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. Users also said that moving this functionality to where it is possible to select the professor’s classes could be clearer. For task 4 (Insert a student to a class as late), users said in a scenario with 100 students, the scroll could be boring and take too much time. Users suggested to lock the input where it could be seen in that view, without having to scroll all the way down. This feedback was taken into account and was implemented in the system.

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 5.12).

Figure 5.13 shows a box plot with the results of SUS questionnaire. 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 1). 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 - figure 5.13 shows exactly that. For example, “I think that I would like to use this system frequently” (Question 1), 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.

## 5.3 Device Fingerprinting Evaluation

In order to catch fraud attendances, a device fingerprint evaluation was also done. A couple of scenarios were planned, put in action and then discussed, according to its results.

### 5.3.1 Method

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.

### 5.3.2 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 practice 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. In the context of these scenarios (that were done under Técnico Lisboa’s network) different IP Addresses do not mean different people, but same IP does mean the same device (which means, same person).

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. Since this comparison in between the same person, the attributes chosen to analyze the current and previous fingerprint are the ones with the lowest similarity ratio \(^2\) (the ones that are more unique from device to device). For example, if the list of fonts (installed) and/or the canvas in different, the system detects and flags the student. 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.

5.4 Discussion
Back in Introduction (chapter 1), it was proposed to create an attendance system, taking into consideration and purpose to mitigate all the problems in this field. The result of this desire is I Am Here!.
To achieve this objective, a website made an easy and cross platform system, so any professor (and student) could use it. Moreover, such application should satisfy the different needs and frustrations of a professor. Within this frame, to evaluate an application such as I Am Here!, both sides of the system were evaluated: the professors’ and the students’. For the students’, it was taken into account its effectiveness, efficiency and error rate when inserting consecutive codes. On the other side (the professors’), it was evaluated according to a set of tasks (meant to cover the general functionalities of the system), were the amount of time users took to complete the tasks and the number of errors were analyzed. Furthermore, since this is an application to be used frequently, in each class, it is important to understand if

\(^2\)https://amiunique.org/fp visited on July 4, 2019
the designed User Interface is easy to learn and use (for both professors and students) and, associated with the tasks, a Single Ease Question (SEQ) and a System Usability Scale (SUS) questionnaires were asked the users to answer.

Overall, the results of the usability user tests show that our application is understandable and user-friendly. 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 an automated way, without taking too much time of the class. The number of errors per class was progressively getting lower in a wave form - because more students attended the Monday’s shift, these students were not as used to the system as the remaining (that went to all classes), which made the Monday’s shift with higher incorrect insertions, creating the wave. But, as explained, the number of incorrect codes was progressively getting lower, having begun in the first class with 58 incorrect codes and, in the last class (3 and a half months later), it ended with only 4.

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. When a student uses a different device, i.e., when someone else is tacking the attendance for a colleague, the system also detects. It is possible to have false positives (when the system flags a student that is in the classroom) - for example, when a student has Indeed a new mobile phone or computer, it is legitimate. Having a short time to insert each code and device fingerprint, I Am Here! can be described as a robust automated attendance control system.
6

Conclusion

Contents

6.1 System Limitations and Future Work ................................. 73
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 this thesis, 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).

It was possible to evaluate the system in a real life scenario - in class - for a complete semester (total of 23 classes). 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. For example, in one of the classes, the system was not used because it was an invited lecture (where a professor was invited to lecture the class) and because it started late, it was used the old-school system of pen and paper. Having the information in paper, the professors wanted to add that to I Am Here! and that was when the functionality to import an attendance was implemented in the system. Along with this, some other functionalities and details were added to the system as well.

From now on, I Am Here! can be used by any professor in Técnico Lisboa. It does not require maintenance or updates to work, as it is running in a loop in a virtual machine in RNL (even if the virtual machine restarts, the server automatically restarts too). The code is public on GitHub, so anyone can use it and/or pick up the work and expand it.

$6.1$ System Limitations and Future Work

Altough the system is working fine with no problems, it has some limitations. It can only be used by professors that have FenixEdu in their faculty, due to having FenixEdu API as the login. Because the system was implemented from scratch, it was not possible to have time to do everything perfectly. The Device Fingerprinting was not much explored because of that.

I Am Here! has plenty of room to expand. 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) would be the next step, as it is important to analyze if different students (from different courses) interact the same way with the system - if the type or length of codes can be the same or if it has to be adapted.

Another important addition could be a step-guide when the user (the professor) enters the first time in the system. A brief (or detailed) explanation of the system, its functionalities and how to use them.

Having graphics could also be helpful to analyze the students attendance tendencies. As the main idea of the system is to track who and how many students are in the classroom, having graphics can

1https://github.com/matildepgrn/I_Am_Here
help to analyze and compare the attendance tendencies between classes.

During the system’s implementation, the professor exported the attendances and published them online so the students could keep track of their attendances. But, if the professor does not publish them, the students do not have a history of their attendances. It could be practical to add the students’ attendance history on the student’s side.

The final addition should be to have/accept more than one attendance in the same class. With Device Fingerprinting and this, the chances of a student cheating the system could probably be very low.
Bibliography


Classroom Usage Evaluation Data
<table>
<thead>
<tr>
<th>Class #</th>
<th># physically attending</th>
<th># registered in the system</th>
<th>Effectiveness (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>49</td>
<td>42</td>
<td>85.7</td>
</tr>
<tr>
<td>2</td>
<td>57</td>
<td>56</td>
<td>98.2</td>
</tr>
<tr>
<td>3</td>
<td>63</td>
<td>57</td>
<td>90.5</td>
</tr>
<tr>
<td>4</td>
<td>56</td>
<td>55</td>
<td>98.2</td>
</tr>
<tr>
<td>5</td>
<td>58</td>
<td>52</td>
<td>89.7</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>63</td>
<td>98.4</td>
</tr>
<tr>
<td>7</td>
<td>55</td>
<td>54</td>
<td>98.2</td>
</tr>
<tr>
<td>8</td>
<td>63</td>
<td>58</td>
<td>92.1</td>
</tr>
<tr>
<td>9</td>
<td>56</td>
<td>56</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>62</td>
<td>55</td>
<td>88.7</td>
</tr>
<tr>
<td>11</td>
<td>53</td>
<td>51</td>
<td>96.2</td>
</tr>
<tr>
<td>12</td>
<td>63</td>
<td>57</td>
<td>90.5</td>
</tr>
<tr>
<td>13</td>
<td>54</td>
<td>54</td>
<td>100</td>
</tr>
<tr>
<td>14</td>
<td>54</td>
<td>50</td>
<td>92.6</td>
</tr>
<tr>
<td>15</td>
<td>48</td>
<td>42</td>
<td>87.5</td>
</tr>
<tr>
<td>16</td>
<td>67</td>
<td>64</td>
<td>95.5</td>
</tr>
<tr>
<td>17</td>
<td>44</td>
<td>46</td>
<td>104.5</td>
</tr>
<tr>
<td>18</td>
<td>54</td>
<td>55</td>
<td>101.9</td>
</tr>
<tr>
<td>19</td>
<td>43</td>
<td>43</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>49</td>
<td>41</td>
<td>83.7</td>
</tr>
<tr>
<td>21</td>
<td>65</td>
<td>65</td>
<td>100</td>
</tr>
<tr>
<td>22</td>
<td>32</td>
<td>32</td>
<td>100</td>
</tr>
<tr>
<td>23</td>
<td>64</td>
<td>62</td>
<td>96.9</td>
</tr>
<tr>
<td>Average:</td>
<td>55.3</td>
<td>52.6</td>
<td>95.1</td>
</tr>
</tbody>
</table>

Table A.1: I Am Here! effectiveness results.
<table>
<thead>
<tr>
<th>Class #</th>
<th>Avg. of codes inserted</th>
<th>Efficiency (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.4</td>
<td>68.1</td>
</tr>
<tr>
<td>2</td>
<td>3.9</td>
<td>75.9</td>
</tr>
<tr>
<td>3</td>
<td>3.3</td>
<td>91.5</td>
</tr>
<tr>
<td>4</td>
<td>3.4</td>
<td>87.9</td>
</tr>
<tr>
<td>5</td>
<td>4.4</td>
<td>67.6</td>
</tr>
<tr>
<td>6</td>
<td>3.8</td>
<td>79.7</td>
</tr>
<tr>
<td>7</td>
<td>3.5</td>
<td>85.3</td>
</tr>
<tr>
<td>8</td>
<td>3.4</td>
<td>88.3</td>
</tr>
<tr>
<td>9</td>
<td>3.7</td>
<td>80.8</td>
</tr>
<tr>
<td>10</td>
<td>3.7</td>
<td>81.0</td>
</tr>
<tr>
<td>11</td>
<td>4.5</td>
<td>67.1</td>
</tr>
<tr>
<td>12</td>
<td>3.4</td>
<td>88.6</td>
</tr>
<tr>
<td>13</td>
<td>3.3</td>
<td>90.5</td>
</tr>
<tr>
<td>14</td>
<td>4.4</td>
<td>68.8</td>
</tr>
<tr>
<td>15</td>
<td>3.7</td>
<td>82.1</td>
</tr>
<tr>
<td>16</td>
<td>3.3</td>
<td>91.1</td>
</tr>
<tr>
<td>17</td>
<td>3.4</td>
<td>88.5</td>
</tr>
<tr>
<td>18</td>
<td>3.2</td>
<td>92.4</td>
</tr>
<tr>
<td>19</td>
<td>3.3</td>
<td>90.2</td>
</tr>
<tr>
<td>20</td>
<td>3.9</td>
<td>75.8</td>
</tr>
<tr>
<td>21</td>
<td>3.8</td>
<td>79.8</td>
</tr>
<tr>
<td>22</td>
<td>3.3</td>
<td>91.7</td>
</tr>
<tr>
<td>23</td>
<td>3.3</td>
<td>91.7</td>
</tr>
<tr>
<td>Average:</td>
<td>3.7</td>
<td>82.8</td>
</tr>
</tbody>
</table>

Table A.2: I Am Here! efficiency results.
<table>
<thead>
<tr>
<th>Class #</th>
<th>Incorrect codes inserted</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>58</td>
</tr>
<tr>
<td>2</td>
<td>48</td>
</tr>
<tr>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>37</td>
</tr>
<tr>
<td>6</td>
<td>36</td>
</tr>
<tr>
<td>7</td>
<td>19</td>
</tr>
<tr>
<td>8</td>
<td>15</td>
</tr>
<tr>
<td>9</td>
<td>18</td>
</tr>
<tr>
<td>10</td>
<td>26</td>
</tr>
<tr>
<td>11</td>
<td>33</td>
</tr>
<tr>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>13</td>
<td>15</td>
</tr>
<tr>
<td>14</td>
<td>27</td>
</tr>
<tr>
<td>15</td>
<td>23</td>
</tr>
<tr>
<td>16</td>
<td>14</td>
</tr>
<tr>
<td>17</td>
<td>9</td>
</tr>
<tr>
<td>18</td>
<td>9</td>
</tr>
<tr>
<td>19</td>
<td>12</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>24</td>
</tr>
<tr>
<td>22</td>
<td>4</td>
</tr>
<tr>
<td>23</td>
<td>4</td>
</tr>
<tr>
<td>Total:</td>
<td>514</td>
</tr>
<tr>
<td>Average:</td>
<td>22.3</td>
</tr>
</tbody>
</table>

Table A.3: I Am Here! error rate results.
System Usability Scale (SUS)
System Usability Scale (SUS) questionnaire

* mandatory

Each question has five response options for respondents: from Strongly Agree (5) to Strongly disagree (1).

1. I think that I would like to use this system frequently. *
2. I found the system unnecessarily complex. *
3. I thought the system was easy to use. *
4. I think that I would need the support of a technical person to be able to use this system. *
5. I found the various functions in this system were well integrated. *
6. I thought there was too much inconsistency in this system. *
7. I would imagine that most people would learn to use this system very quickly. *
8. I found the system very cumbersome to use. *
9. I felt very confident using the system. *
10. I needed to learn a lot of things before I could get going with this system. *
11. I think the functionalities of the system are enough for a professor. *
12. I wanted to do something but I did not find the way to do it. *
Single Ease Question (SEQ)
Single Ease Question (SEQ) questionnaire
* mandatory

1. Task: *
   - 1
   - 2
   - 3
   - 4
   - 5
   - 6
   - 7

2. How difficult or easy did you find the task? *
   
   1 2 3 4 5 6 7
   Very Hard
   Very Easy
Single Ease Question (SEQ) results
<table>
<thead>
<tr>
<th>User / Question</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>18</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Avg.</td>
<td>4,8</td>
<td>1,4</td>
<td>4,7</td>
<td>1,3</td>
<td>4,7</td>
<td>1,3</td>
<td>4,4</td>
<td>1,2</td>
<td>4,2</td>
<td>1,9</td>
<td>4,9</td>
<td>2,3</td>
</tr>
</tbody>
</table>

Table D.1: System Usability Scale answers by each user.
User Evaluation Evaluation Data

<table>
<thead>
<tr>
<th>Task</th>
<th>Minimum</th>
<th>Lower Quartile</th>
<th>Upper Quartile</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>32</td>
<td>45,8</td>
<td>59</td>
<td>70</td>
</tr>
<tr>
<td>Task 2</td>
<td>40</td>
<td>45,5</td>
<td>52,8</td>
<td>63</td>
</tr>
<tr>
<td>Task 3</td>
<td>12</td>
<td>19,8</td>
<td>24,5</td>
<td>45</td>
</tr>
<tr>
<td>Task 4</td>
<td>23</td>
<td>43,8</td>
<td>62,1</td>
<td>210</td>
</tr>
<tr>
<td>Task 5</td>
<td>51,4</td>
<td>79</td>
<td>104,2</td>
<td>230</td>
</tr>
<tr>
<td>Task 6</td>
<td>8,0</td>
<td>33,3</td>
<td>40,5</td>
<td>51</td>
</tr>
<tr>
<td>Task 7</td>
<td>38,3</td>
<td>70</td>
<td>85,3</td>
<td>230</td>
</tr>
</tbody>
</table>

Table E.1: Time (in seconds) to complete each task: minimum value, lower quartile, upper quartile and maximum
### Table E.2: Errors during each task: minimum value, lower quartile, upper quartile and maximum

<table>
<thead>
<tr>
<th>Task</th>
<th>Minimum</th>
<th>Lower Quartile</th>
<th>Upper Quartile</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Task 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Task 3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Task 4</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Task 5</td>
<td>1,3</td>
<td>3,5</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>Task 6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Task 7</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>

### Table E.3: Single Ease Question (SEQ) answers for each task: minimum value, lower quartile, upper quartile and maximum

<table>
<thead>
<tr>
<th>Task</th>
<th>Minimum</th>
<th>Lower Quartile</th>
<th>Upper Quartile</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Task 2</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Task 3</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Task 4</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Task 5</td>
<td>1</td>
<td>1</td>
<td>1,3</td>
<td>2</td>
</tr>
<tr>
<td>Task 6</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Task 7</td>
<td>1</td>
<td>1</td>
<td>2,8</td>
<td>3</td>
</tr>
</tbody>
</table>

Table E.2: Errors during each task: minimum value, lower quartile, upper quartile and maximum

Table E.3: Single Ease Question (SEQ) answers for each task: minimum value, lower quartile, upper quartile and maximum
Step by Step: how to update HTTPS certificates
How to update HTTPS certificates for *I Am Here*

By: Matilde Paraíso Nascimento

Although the Node.js projects supports HTTPS, because the virtual machine where the system is hosted also hosts Moodle, the Node.js system is running behind an Nginx reverse proxy. HTTPS is configured for Nginx (the configuration for the reverse proxy virtual host is in the following file: /etc/nginx/sites-enabled/iamhere_proxy).

1. Go to [SSL For Free website](https://sslforfree.com).
2. Type the correct domain (classcheck.tk) and select “Create Free SSL Certificate”.


4. Click “Manually Verify Domain”.

5. This will show up. **Do not click on the links in 3 yet, or you will have to wait for 1 hour!**
1. Go into the DNS management page that your domains use (This link may help with setting up your TXT records [ignoring Google specific parts]).

2. Add the following TXT records below to the DNS server for each domain (Please note your DNS software may auto-add the domain in the name field, contact DNS provider if unsure or if you get NXDOMAIN errors):

   1. Add TXT record with the name/host

   ```
   _acme-challenge.classcheck.tk
   ```

   with the value

   ```
   cdqcKervVogNdhimVuVCfahJd1URxejDFys.Jsl-M7WZU
   ```

   and a TTL (Time to Live) (in seconds) of

   ```
   1
   ```

   2. Add TXT record with the name/host

   ```
   _acme-challenge.www.classcheck.tk
   ```

   with the value

   ```
   VcyaZ2Mufct-T9jW91OdSogm6cOX8dQrl.Jgmxyx1Asog
   ```

   and a TTL (Time to Live) (in seconds) of

   ```
   1
   ```

3. Verify TXT records have been propagated by going to the following links. The corresponding values above should show up within the record:

   1. Verify _acme-challenge.classcheck.tk
   2. Verify _acme-challenge.www.classcheck.tk

4. You may have to wait a minute to a couple minutes for the DNS TXT record to propagate if the TTL record value does not show up or is not the same as the above values. If you get an error during verification that says "JWST has invalid anti-replay nonce" then just refresh the page (resending post data if prompted) until it works.

5. Click Download SSL Certificate below.

   ![Download SSL Certificate](image)

   - I Have My Own CSR

6. Go to **Dot TK DNS Management** for the right domain (classcheck.tk). This will show up (the values may be different):

   ![Modify Records](image)

   ![Add Records](image)
7. Put the given value into the respective target. The end result should be:

8. Click “Save Changes” on the bottom right of the modify records table. You should see this feedback:
   - Record modified successfully
   - Record modified successfully

9. Wait 2-3 minutes.
10. Now, you can check the links mentioned in point 5, to verify the TXT (the value for the TXT should match the inserted one). If not, check if the values in My Freenom are correct, and wait 1 hour before repeating this step.
11. Click “Download SSL Certificate” (check figure from point 5).
12. You should three boxes: Certificate, Private Key, CA Bundle. Save the content of each box to the following files:

<table>
<thead>
<tr>
<th>Box</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td>Certificate</td>
<td>cert.crt</td>
</tr>
<tr>
<td>Private Key</td>
<td>cert.key</td>
</tr>
<tr>
<td>CA Bundle</td>
<td>cert_ca.crt</td>
</tr>
</tbody>
</table>

`cert_bundle.crt` : cert + ca
`cert.key` : private key

13. Now, you need to copy the files to `/home/iamhere/I_Am_Here/ssl` in the virtual machine hosting the server. You can use the `scp` command.
14. In the server, after sending the files to the correct directory, to create `cert_bundle.crt`, run:

```
$ cd /home/iamhere/I_Am_Here/ssl
$ ( cat cert.crt; echo; cat cert_ca.crt ) > cert_bundle.crt
```

15. Now, you can restart Nginx. Run (as `root`):

```
$ /etc/init.d/nginx restart
```

16. You are done. Try to access the system via HTTPS.

For more info, check the SSL For Free website.