



# **CROSS City Mobile Application: Gamified Peer-Based Location Certification**

**Ricardo António Augusto Grade**

Thesis for obtaining a Master of Science Degree in

**Computer Science and Engineering**

Supervisors: Prof. Miguel Filipe Leitão Pardal

Dr. Samih Eisa Suliman Abdalla

## **Examination Committee**

Chairperson: Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur

Supervisor: Prof. Miguel Filipe Leitão Pardal

Member of the Committee: Prof. Hugo Miguel Aleixo Albuquerque Nicolau

**November 2022**



## Acknowledgments

I want to thank my coordinators, Miguel Pardal and Samih Eisa, for the helpful, wise, and fruitful discussions we had together. I want to thank my colleagues and friends, Rafael Figueiredo and Lucas Vicente, with whom I have discussed the difficulties that have arisen along the way, helping me to overcome them, creating an environment of mutual support. I want to thank my family, especially my mother, father and sister, for providing the environment where I have the space to focus on my work and fulfill my ambitions. Finally, I want to thank my best friend, Sara Machado, for her personal support during the development of my dissertation, boosting my professional focus. Without all of them, this work would not have happened.

Furthermore, this work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UIDB/50021/2020 (INESC-ID) and through project with reference PTDC/CCI-COM/31440/2017 (SureThing).



## Abstract

There are many cities around the world with iconic sites such as viewpoints, museums and historical monuments. Nowadays, tourists turn to digital platforms to discover new places to explore. *CROSS City* is a smart tourism mobile application that enhances the users visiting experience by rewarding them for carrying out tourist routes. From a technical standpoint, CROSS City certifies the user location when visiting points of interest, resorting to location certification strategies, that take advantage of both the diversity of the existing Wi-Fi network infrastructure throughout the city, as well as the presence of other users on-site. This work extended the previous app prototype with a new peer-based location certification strategy. Moreover, gamification elements were added to further encourage user participation. This work was evaluated both in laboratory experiments and with users in a real-world scenario – a university campus tour – which demonstrated that the new peer-based strategy is both feasible and collusion-resilient.

**Keywords:** Location Certification, Location Spoofing Prevention, Peer-to-Peer Communication, Bluetooth, Internet of Things, Gamification, User Experience



## Resumo

Existem muitas cidades ao redor do mundo repletas de locais emblemáticos como miradouros, museus e monumentos históricos. Hoje em dia, os turistas recorrem a plataformas digitais para descobrir novos locais para explorar. O *CROSS City* é uma aplicação móvel de turismo inteligente que aumenta a experiência de visita dos utilizadores recompensando-os pela realização de rotas turísticas. Do ponto de vista técnico, o *CROSS City* certifica a localização do utilizador ao visitar pontos de interesse, recorrendo a estratégias de certificação de localização, que se aproveitam da diversidade da infraestrutura de rede Wi-Fi existente por toda a cidade, bem como da presença de outros utilizadores que se encontrem nos locais. Este trabalho estendeu a aplicação protótipo anterior com uma nova estratégia de certificação de localização baseada em pares. Foram também adicionados elementos de gamificação destinados a incentivar a participação de utilizadores. Este trabalho foi avaliado tanto em experiências de laboratório quanto com utilizadores num cenário do mundo real – um circuito pelo campus universitário – os quais demonstraram que a nova estratégia baseada em pares é viável e resiliente contra conluio.

**Palavras-chave:** Certificação de Localização, Prevenção da Falsificação de Localização, Comunicação Ponto-a-Ponto, Bluetooth, Internet das Coisas, Gamificação, Experiência de Utilizador





# Contents

Acknowledgments . . . . .	iii
Abstract . . . . .	v
Resumo . . . . .	vii
List of Tables . . . . .	xiii
List of Figures . . . . .	xv
Acronyms . . . . .	xix
<b>1 Introduction</b>	<b>1</b>
1.1 Contributions . . . . .	2
1.2 Outline . . . . .	2
<b>2 Background and Related Work</b>	<b>5</b>
2.1 CROSS City Prototype . . . . .	5
2.2 Location Certification Techniques . . . . .	7
2.2.1 Infrastructure-Based Techniques . . . . .	7
2.2.1.1 Wi-Fi Techniques . . . . .	8
2.2.1.2 Kiosk Techniques . . . . .	8
2.2.2 Peer-Based Techniques . . . . .	9
2.2.2.1 Ensuring Nontransferability of Endorsements . . . . .	10
2.2.2.2 Ensuring Unforgeability of Endorsements . . . . .	11
2.2.2.3 Preventing Prover-Witness Collusion . . . . .	11
2.2.2.4 Preventing Prover-Prover Collusion . . . . .	12

2.2.2.5	Preserving User Location Privacy . . . . .	12
2.3	User Experience . . . . .	13
2.3.1	User Interface . . . . .	13
2.3.2	Empirical User Experience Evaluation . . . . .	14
2.3.3	User Experience in Mobile Augmented Reality . . . . .	15
2.3.4	User Experience in Location Certification Systems . . . . .	16
2.4	Gamification . . . . .	16
2.4.1	Gamification in Tourism . . . . .	17
2.4.2	Gamification of an Engineering Course . . . . .	18
2.4.3	Gamification-Based Incentive Mechanism . . . . .	18
2.4.4	Gamification of a Location-Based Mobile Application . . . . .	19
2.5	Summary . . . . .	20
<b>3</b>	<b>Implementation Preliminaries</b>	<b>23</b>
3.1	Architecture Overview . . . . .	23
3.2	CROSS City Re-Implementation . . . . .	24
3.2.1	Server . . . . .	25
3.2.2	Contract . . . . .	27
3.2.3	Mobile Application . . . . .	27
3.3	Summary . . . . .	31
<b>4</b>	<b>Implementation</b>	<b>33</b>
4.1	Assumptions . . . . .	33
4.2	Guidelines . . . . .	33
4.3	P2P Witnessing Strategy . . . . .	34
4.3.1	Threat Model . . . . .	34
4.3.2	Peer Endorsement Acquisition . . . . .	35
4.3.2.1	Short-Range Wireless Technology . . . . .	35

4.3.2.2	Peer Endorsement Acquisition Protocol . . . . .	36
4.3.2.3	Distance-Bounding Protocol . . . . .	40
4.3.3	Peer Endorsement Validation . . . . .	41
4.3.3.1	Peer Endorsement Validation Process . . . . .	41
4.3.3.2	Witness Decay . . . . .	43
4.3.3.3	Visit Confidence . . . . .	44
4.3.4	Threat Mitigation Assessment . . . . .	46
4.4	Gamification . . . . .	47
4.5	Summary . . . . .	51
<b>5</b>	<b>Evaluation</b>	<b>53</b>
5.1	P2P Witnessing Strategy Experiments . . . . .	53
5.1.1	Peer Endorsement Validation Assessment . . . . .	53
5.1.1.1	Parameter Values Estimation . . . . .	54
5.1.1.2	Prover-Witness Collusion-Resilience . . . . .	55
5.1.1.3	Validation Protocol Responsiveness . . . . .	56
5.1.2	Peer Endorsement Acquisition Assessment . . . . .	59
5.1.2.1	Prover-Prover Collusion-Resilience . . . . .	59
5.1.2.2	Acquisition Protocol Performance . . . . .	63
5.1.2.3	Protocol Impact on the Mobile Device Power Resources . . . . .	64
5.2	Evaluation with Users . . . . .	65
5.2.1	Instrumentalization Metrics Analysis . . . . .	66
5.2.2	Gamification Impact . . . . .	69
5.3	Summary . . . . .	71
<b>6</b>	<b>Conclusion</b>	<b>73</b>
6.1	Achievements . . . . .	73
6.2	Future Work . . . . .	74

<b>Bibliography</b>	<b>77</b>
<b>A CROSS City Mobile Application Screens</b>	<b>81</b>
<b>B CROSS City Future Extensions</b>	<b>91</b>
<b>C CROSS City Campus Tour - Google Forms</b>	<b>97</b>

# List of Tables

- 2.1 Properties of peer-based location certification systems. . . . . 21
  
- 5.1 Sum of the weights of endorsements that witnesses issue, with variant cardinality and reputation (through previously successfully validated visits); the colors are used just to differentiate the minimum reputation for the prover to successfully validate the visit . . . . . 54
- 5.2 Number of collusions possible to perform with the specified number of newly created witnesses. . . . . 56
- 5.3 Average ping time per challenge statistics; Pings from Cascais, Portugal to the mentioned locations. . . . . 62
- 5.4 Witness acceptance rate of honest and colluding provers claims, given the threshold 34.5 milliseconds (explained below how it was defined). . . . . 62
- 5.5 Distribution of the different components of the protocol execution time (connection establishment time, connection setup time, and endorsement time). . . . . 64
- 5.6 Metrics of the route through the Alameda campus with 3 users; the users are arranged in the ascending temporal order of their visit submission; the colors assigned to each user serve solely to better distinguish the respective rows. . . . . 66
- 5.7 Metrics of the route through the Alameda campus with 7 users. . . . . 67



# List of Figures

- 2.1 Overview of the CROSS City prototype architecture. . . . . 6
- 2.2 Wi-Fi-based location certification systems overview. . . . . 8
- 2.3 Peer-based location certification systems overview. . . . . 9
  
- 3.1 Extended architecture overview of CROSS City. . . . . 23
- 3.2 Server database entities and relationships between them. . . . . 26
- 3.3 Mobile application database entities and relationships between them. . . . . 29
  
- 4.1 Prover-witness message flow . . . . . 39
- 4.2 Displacement confidence decline according to travel speed. . . . . 45
- 4.3 Confidence threshold adjustment according to the prover reputation. . . . . 46
- 4.4 All-time scoreboard. . . . . 49
- 4.5 Earned badges. . . . . 49
- 4.6 In-app card for gem-based payment. . . . . 50
  
- 5.1 This plot was generated attempting to perform multiple collusions with 10 newly created witnesses, we can see that even with 10, only 2 collusions are possible to perform. . . . . 56
- 5.2 Visit validation time varying the number of witnesses who endorse the visit. . . . 57
- 5.3 Visit validation time varying the number of simultaneous visit submissions. . . . 58
- 5.4 Average challenge response times with the devices 1 meter apart. . . . . 60
- 5.5 Average challenge response times with the devices 10 meters apart. . . . . 60
- 5.6 Average challenge response times with the devices in adjacent rooms. . . . . 60

5.7	Frequency of average challenge response times. . . . .	61
5.8	Execution time of the endorsement acquisition protocol on Samsung Galaxy S9. .	63
5.9	Execution time of the endorsement acquisition protocol on Xiaomi POCO X3 Pro.	63
5.10	Location map of points of interest that compose the tour route. . . . .	65
5.11	Group photo on the tour. . . . .	65
A.1	Navigation drawer with app menus. . . . .	82
A.2	List of untraveled routes. . . . .	82
A.3	Route yet untraveled. . . . .	83
A.4	Point of interest yet untraveled. . . . .	83
A.5	List of routes already initiated. . . . .	84
A.6	Route already initiated. . . . .	84
A.7	Start of visit. . . . .	85
A.8	Feedback on evidence collected during the visit. . . . .	85
A.9	Opening visit rewards chest. . . . .	86
A.10	Visit rewards. . . . .	86
A.11	Rewards notification. . . . .	87
A.12	User profile. . . . .	87
A.13	Badge yet unearned. . . . .	88
A.14	Badge earned. . . . .	88
A.15	Choice of location certification strategies scheme to be used in a visit. . . . .	89
B.1	List of untraveled routes. . . . .	92
B.2	List of traveled routes. . . . .	92
B.3	Route yet untraveled. . . . .	93
B.4	Route already traveled. . . . .	93
B.5	Multiple choice challenge. . . . .	94
B.6	Challenge of identifying the correct image. . . . .	94



B.7	Marker hunt challenge. . . . .	95
B.8	How far is the user from the point of interest. . . . .	95
B.9	Shortest path of a route. . . . .	96



# Acronyms

AP Access Point

API Application Programming Interface

APK Android Package Kit

BLE Bluetooth Low Energy

CA Certificate Authority

DB Distance-Bounding

GNSS Global Navigation Satellite System

GPS Global Positioning System

HTTP Hypertext Transfer Protocol

I/O Input and Output

IoT Internet of Things

IT Information Technology

JDBC Java Database Connectivity

JWT JSON Web Token

LBS Location-Based Service

LCA Location Certification Authority

LPS Location Proof Server

MAC Media Access Control

MTU Maximum Transmission Unit

OS Operating System

P2P Peer-to-Peer

PoI Point of Interest

REST REpresentational State Transfer

RSA Rivest–Shamir–Adleman

RTT Round Trip Time

SSID Service Set Identifier

TLS Transport Layer Security

# Chapter 1

## Introduction

Cities are rich in points of interest, so much so that tourists often do not know which sites to visit. Popular platforms like Lonely Planet <sup>1</sup> or Tripadvisor <sup>2</sup> provide access to a wealth of information where users can look for new places to explore. However, there may be a lack of motivation to visit some places, especially those further away from the city centre. Offering cultural, historical or gastronomic routes and rewarding users for carrying them out is a way of engaging and motivating tourists and broaden their visiting experience across the city. *CROSS* <sup>3</sup> *City* <sup>4</sup> is a smart tourism mobile application (henceforth abbreviated as *app*) that rewards users for actually visiting all the points of interest that make up a tourist route.

**Location Certification** Since any malicious user could spoof their geocoordinates, the app validates the user visits by resorting to location certification strategies. All location certification strategies developed in the CROSS City app prototype [MCP20] only worked in locations with the necessary surrounding Wi-Fi network infrastructure in place. Moreover, location evidence relied on the reporting of volatile Wi-Fi networks to obtain location certification with temporal granularity, which could only be validated after a full period [CEP22], average case 12 hours, worst case 24 hours. Finally, the different types of location evidence developed are not linked to the user who captures it, as such it allows evidence transfer between users. This work addressed these limitations by introducing a new peer-based collusion-resilient location certification strategy. The insight was to leverage the moments when user devices *see* each other to produce co-located location evidence in places with no surrounding infrastructure that can be validated as soon as submitted.

---

<sup>1</sup><https://www.lonelyplanet.com/>

<sup>2</sup><https://www.tripadvisor.com/>

<sup>3</sup>location proof techniques for consumer mobile applications

<sup>4</sup><https://youtu.be/-Ev1JLfb7W0>

**User Participation Need** The main goal of the CROSS City app prototype was to prevent location spoofing attacks, thereby demonstrating the feasibility of its location certification strategies, so usability and engagement were secondary and therefore not taken into account. However, since the location certification strategy just introduced is peer-based, user participation is a critical factor for its proper functioning. With that in mind, we took all the lessons from the prototype and fully re-implemented the app to deliver a better user experience, with gamification elements to encourage users to further use the app and embrace the new peer-based strategy.

**Evaluation** This work was evaluated both in laboratory experiments and in a real-world setting. In the laboratory experiments we demonstrated that the peer endorsement acquisition and validation protocols are effective, responsive and collusion-resilient. For the real-world assessment, we organized a university campus tour with real users, where we demonstrated that the peer-based strategy promises to be feasible in a real-world setting, and the built-in gamification elements served their purpose, encouraging users to further use the app and adopt the new strategy.

## 1.1 Contributions

This work delivers a new version of CROSS City where:

- **Main Contribution:** We developed a new peer-based collusion-resilient location certification strategy, capable of validating the user location in crowded places where there is no surrounding infrastructure, which was not possible with the previous strategies;
- **Auxiliary Contribution:** We added gamification elements into the app to further engage and encourage users to use it, rewarding more generously those users who adopt the new peer-based strategy.

Furthermore, this work was articulated with two others: *CROSS City Cloud* [MP22a] that enables the processing and storage of CROSS City data in a cloud infrastructure; as well as *SureRepute* [MP22b] which manages the reputation of the app users.

## 1.2 Outline

The remainder of this document is structured as follows: in Chapter 2, we present the background and related work developed in the areas of location certification techniques, user experience and

gamification; in Chapter 3, we present the preliminaries for the implementation of this work, including the extended architecture and the re-implementation of the system; in Chapter 4, we present the implementation of the solution, more specifically of the new peer-based collusion-resilient location certification strategy and the new gamification elements; in Chapter 5, we present the evaluation both in laboratory experiments and in a real-world scenario with real users; in Chapter 6, we present the conclusion and indicate possible extensions for the future.

Finally in Appendix A, we present some screens of the new CROSS City app, in Appendix B the screen mockups of the possible future extensions, and in Appendix C the form that was given to users to measure the gamification impact.





## Chapter 2

# Background and Related Work

In this Chapter, we begin by presenting the prototype of the CROSS City system that was extended. We then present background and related work on location certification techniques, as we introduce a new peer-based collusion-resilient location certification strategy into the system. Finally, we present relevant concepts of user experience and gamification, to engage and motivate users, as the peer-based strategy relies on user participation.

### 2.1 CROSS City Prototype

The previous version of CROSS City [MCP20] was a prototype system with a mobile application that is able to collect evidence of where its users are. The users are tourists that travel around the city and perform tourist routes that are made up of points of interest and, upon validation of the location evidence collected along the way, they are rewarded with coupons or souvenirs.

The entities that make up the system are illustrated in Figure 2.1. The client is an Android <sup>1</sup> app that keeps the catalog <sup>2</sup> in cache, collects location evidence, stores it in a queue to be submitted to the server, and presents the user interface to the tourist. The server validates the location evidence presented by the clients and verifies their validity. Also, it maintains the user information and the catalog, and provides a set of services that are available through an API that clients use to communicate with the server.

---

<sup>1</sup><https://www.android.com/what-is-android/>

<sup>2</sup>Information on the locations of the points of interest that make up each tourist route and their respective rewards.

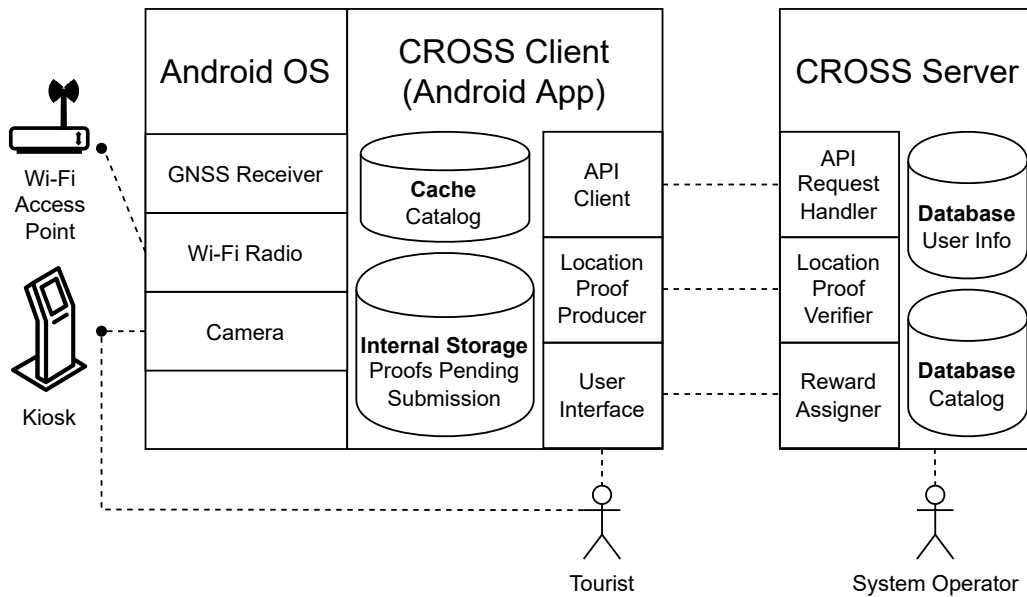


Figure 2.1: Overview of the CROSS City prototype architecture.

The three location certification strategies presented in the original system are:

- *Wi-Fi Scavenging*: The app captures the SSIDs of the Wi-Fi networks it detects, which are constantly being broadcast, and timestamps them. At the end of the visit, the user submits the evidence and the server checks it against its database of access points present at the location of that visit. This strategy is inconspicuous and cost-effective, as it leverages the infrastructure present at each point of interest, however, once an opponent knows the list of visible Wi-Fi networks at a given location, they can forge location evidence;
- *TOTP*: It works the same as the previous one from the client point of view, however, some APs are modified to broadcast a sequence of identifiers that change periodically and are apparently random, but which the server knows. This strategy allows the server to know how long the user has been in a certain location, analyzing the different SSIDs that were submitted;
- *Kiosk*<sup>3</sup>: The user must interact with a kiosk (e.g. a ticket vending machine) that acts as a witness. This strategy is the most secure of all, especially if the kiosk is monitored by an employee, however, it is also the least convenient for the user, as it requires them to explicitly interact with the kiosk.

<sup>3</sup>The Kiosk strategy was only implemented in a later work, namely, SurePresence [Fra21].

## 2.2 Location Certification Techniques

GPS is the most widely used system by mobile applications that need to obtain the user location to provide certain services. It is convenient for developers and users, and it has worldwide coverage and support, such as assisted GPS to improve startup performance by picking up signals from the surrounding infrastructure. However, the GPS coordinates obtained can be easily spoofed at both the OS and application level [NSY<sup>+</sup>20], hence the need to verify the location that users report on systems that reward users based on those locations. Mobile devices are increasingly equipped with various network interfaces, such as Wi-Fi and Bluetooth, which can be used to prove that the device is in a certain location, through detection of the surrounding context and in cooperation with nearby devices.

We define the following *entities* that make up these systems:

- *Prover*: Claims to be at a certain location at a certain time and subsequently intends to prove its authenticity;
- *Witness*: Is at the same location as the prover and can endorse claims;
- *Verifier*: Validates, and certifies accordingly, claims supported by endorsements.

We define the following *terms* to homogenize the technical vocabulary to be used:

- *Claim*: Document that proves produce (usually signed) for being at a certain location at a certain time;
- *Endorsement*: Evidence that witnesses issue (usually signed) endorsing a claim;
- *Certificate*: Certification that verifiers issue (usually signed) attesting a claim that is supported by sufficient valid endorsements.

*Endorsement schemes* can be categorized into two groups [NSY<sup>+</sup>20]:

- *Infrastructure-Based*: Where the witnesses are elements of the surrounding reliable wireless infrastructure;
- *Peer-Based*: Where the witnesses are other on-site users. This approach is particularly useful in crowded locations where wireless infrastructure is insufficient.

### 2.2.1 Infrastructure-Based Techniques

These techniques take advantage of the wireless infrastructure that is often already in place at the locations. This kind of infrastructure is being increasingly deployed in smart cities with the expansion of the IoT.

Next, we present techniques based on wireless infrastructure, namely, Wi-Fi APs and kiosks.

### 2.2.1.1 Wi-Fi Techniques

The overview of systems that employ Wi-Fi-based techniques for location certification is illustrated in Figure 2.2. First, the user captures the SSIDs of the Wi-Fi networks that are constantly being broadcast along the way. Second, the user submits the endorsements, consisting of the detected SSIDs and their detection time, to the verifier, which in turn validates them aided by a database of the locations of these APs.

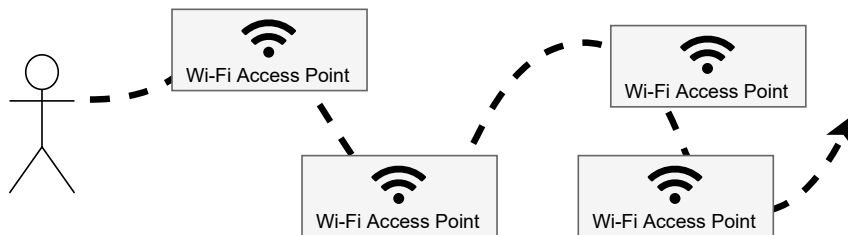


Figure 2.2: Wi-Fi-based location certification systems overview.

Alamleh et al. [AA20] proposed a cheat-proof system to validate the GPS coordinates that users report via time-sensitive location validation data that is broadcast from nearby Wi-Fi APs. This system uses APs whose virtual interface has been configured and programmed to broadcast location validation data in their SSID, which changes unpredictably to an observer, but which can be calculated by the server. However, this system requires that the clocks of the APs are properly synchronized with the server clock, which can, nevertheless, be difficult to guarantee in a distributed system. Over the course of a session, mobile devices collect location validation data. This data is collected from the APs with the highest signal strength that have a MAC address registered as a participating AP in the system.

These techniques, with no user device interaction required, are preferable to the user in terms of privacy, as they minimize their exposure to attacks and ensure that only the verifier can determine the locations the user has walked through.

### 2.2.1.2 Kiosk Techniques

In systems that employ kiosk-based techniques for location certification the user interacts with existing kiosks, along the way, to acquire endorsements.

SurePresence [Fra21] is a system that allows users to certify their location using wearable devices and kiosks. For this purpose, it has developed three techniques:

- *Kiosk-Only*: The user inserts the citizen card into the kiosk reader, which retrieves basic information (including card identifier and name) that is not protected and therefore does not need to be unlocked by the authentication PIN. The kiosk issues the endorsement and sends it to the verifier;
- *Kiosk-Wearable*: The wearable establishes a BLE connection with the kiosk, enabling device scanning at the kiosk and selecting the identifier of the device. The wearable sends the signed user information (including the location and the email the user is logged in to) to the kiosk, which issues the endorsement and sends it to the verifier;
- *Kiosk-Smartphone*: In the first phase, the smartphone app generates and displays a QR code (which contains the signed claim) which is scanned by the kiosk. In the second phase, the kiosk generates and displays another QR code (which contains the signed endorsement) that is scanned by the smartphone app. In the end, unlike the other techniques, it is the app that sends the endorsement to the verifier. Thus, this technique works even in scenarios where the kiosk does not have Internet access.

## 2.2.2 Peer-Based Techniques

These techniques leverage the existence of on-site users willing to endorse the location of their peers, issuing them endorsements via Bluetooth or another short-range wireless communication protocol. The overview of these systems is illustrated in Figure 2.3, operating in two phases:

1. *Endorsement Acquisition*: The prover broadcasts claims and collects endorsements issued by nearby witnesses;
2. *Claim Validation*: The verifier verifies the validity of the endorsements supporting the claim.

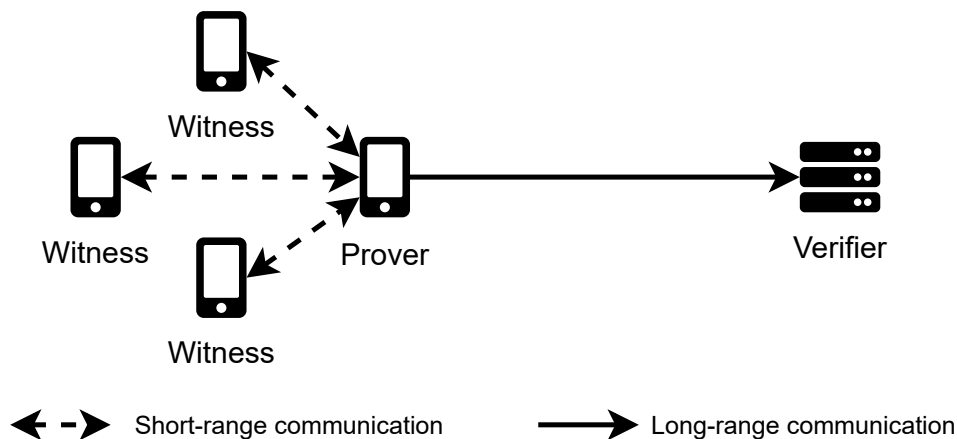


Figure 2.3: Peer-based location certification systems overview.

SureThing [FP18], LINK [TCB10], APPLAUS [ZC11], CREPUSCOLO [CCCDP13] and PASPORT [NSY<sup>+</sup>20] are peer-based location certification systems.

These systems propose extra entities with different functions: LINK [TCB10] contains the LCA (Location Certification Authority), which receives endorsements and presents the LBS (Location-Based Service) with the decision to accept or reject the associated claim based on their validity. APPLAUS [ZC11] and CREPUSCOLO [CCCDP13] contain the LPS (Location Proof Server), which maintains location certificates, although it is considered unreliable; SureThing [FP18], APPLAUS [ZC11] and CREPUSCOLO [CCCDP13] contain a CA (Certificate Authority) for signing public key certificates. In addition, in APPLAUS [ZC11] and CREPUSCOLO [CCCDP13], the CA is also responsible for the anonymity of users, managing the mapping between the pseudonyms that it assigns to them and their real identity.

These systems can suffer the following *attacks*:

- *Endorsements Transfer*: A malicious prover uses endorsements issued to another;
- *Endorsements Forging*: A malicious user can forge endorsements either for their own benefit or to defame another;
- *Prover-Witness Collusion*: When the prover and witnesses are malicious and those witnesses endorse the false claim of that prover;
- *Prover-Prover Collusion*: When two provers are malicious and the first channels a claim to the second who is at a different location, which in turn broadcasts it over Bluetooth, causing honest witnesses to endorse the false claim of the first;
- *User Location Privacy Breach*: Users often broadcast their identity to neighboring devices or a third-party server, however if the user identity is publicly disclosed, their personal information would be leaked.

Next, we present defense mechanisms against these attacks that systems implement.

### 2.2.2.1 Ensuring Nontransferability of Endorsements

So that endorsements cannot be used to endorse another claim they do not refer to, the systems implement protocols based on random or sequence numbers.

In LINK [TCB10], all endorsements have the sequence number associated with their claim built in, as such, they cannot be associated with a different claim. Also, as the LCA stores the latest sequence number, the claim cannot be resubmitted, as it would be able to verify that it has already processed this claim.

In both APPLAUS [ZC11] and CREPUSCOLO [CCCDP13], each claim contains an associated nonce (number used only once) that is present in all endorsements issued for this claim, therefore, they could not be reused to endorse a different one, and the existence of two claims with the same nonce would reveal its reproduction.

In PASPORT [NSY<sup>+</sup>20], two secret random numbers are generated for a claim, which are used to respond to the challenges that are part of the performed Distance Bounding (DB) protocol. When submitting endorsements the prover has to include these two secret random numbers, therefore, no other prover can include them. Otherwise, if the prover were to provide these secret numbers to others, they would risk being impersonated.

In addition, endorsements cannot be withheld for later submission either, in order to simulate that they refer to a claim issued at a later time, since each endorsement bears the timestamp corresponding to the moment it was issued.

### **2.2.2.2 Ensuring Unforgeability of Endorsements**

To avoid eavesdropping on messages, provers and witnesses encrypt them with the public key of the verifier (or LPS, regarding the systems that keep location certificates there), not only providing eavesdropping protection, but also hiding their content from other users. To avoid tampering with messages, users sign them with their private key, or, in the case of systems that use pseudonyms, with the private key associated with the pseudonym they selected, so as not to reveal their identity.

### **2.2.2.3 Preventing Prover-Witness Collusion**

To prevent provers from colluding with witnesses, the systems implement witness redundancy and decay and reputation-based mechanisms, otherwise malicious witnesses could systematically corroborate false claims of other users.

SureThing [FP18] and LINK [TCB10] devalue endorsements issued by the same witnesses to the claims of a given user, as they assign weighted reputations to witnesses, which are lower depending on the number of times they have endorsed claims of that user. In LINK [TCB10], to remember the number of times a user acted as a witness on behalf of another, the LCA maintains a matrix. In addition, the LCA takes into account the location and timestamp present on the last valid claim submitted and finds out whether it is physically possible for such a displacement to occur.

In both APPLAUS [ZC11] and CREPUSCOLO [CCCDP13], they check how many other co-located and concurrent endorsements exist. To do this, the LPS takes advantage of knowing which pseudonyms are at each location at a given time, to establish a pseudonyms map. For example, say a prover with pseudonym  $Pa$  submits only a singular endorsement issued by a witness with pseudonym  $Pb$ , and the LPS knows from the pseudonyms map that there are more users in that region who could also have issued endorsements, the  $Pa$  and  $Pb$  users are considered potential colluders and their reputation is diminished.

In PASPORT [NSY<sup>+</sup>20], the verifier selects, from the range of witnesses that are in the same region as the prover, a subset of them who will be able to endorse a given claim, in this way, the prover would not only need to collude with a sufficient number of witnesses to mislead the verifier, but would also need those to be selected, which would hardly happen systematically, as the verifier selects the witnesses according to an entropy model.

These mechanisms focus on preventing users from systematically colluding, yet are vulnerable to attacks in which a particular user behaves correctly and occasionally misbehaves. Furthermore, they can hardly identify a massive pool of malicious users, as they disguise themselves.

#### **2.2.2.4 Preventing Prover-Prover Collusion**

CREPUSCOLO [CCCDP13] circumvents this attack by adding *Token Providers* to the infrastructure that act as trusted witnesses located in known locations that generate tokens which contain images from surveillance cameras, allowing the verifier to verify whether who was in that location was the prover to which the endorsement refer or someone else impersonating them.

The issuance of endorsements in PASPORT [NSY<sup>+</sup>20] is carried out based on a DB protocol, where the maximum RTT between the prover and the witness is calculated based on the acceptable distance between them. The witness sends challenges that only the legitimate prover can respond to, so if a malicious prover is tunneling the communication between the legitimate prover and the witness, when the endorsement is submitted the verifier can detect that the maximum RTT has been exceeded and therefore rejects it.

#### **2.2.2.5 Preserving User Location Privacy**

The privacy of the user location is critical for systems that rely on user participation as this is considered a sensitive asset, since if they believe their location is not being kept private they might not join the system.



In APPLAUS [ZC11], a set of pseudonyms is assigned to each user, when a prover intends to generate a claim, they select a pseudonym taking into account a Poisson distribution model, so that no one can statistically correlate it with the other pseudonyms of the prover, in order to protect the location history of users. In addition, CREPUSCOLO [CCCDP13] goes one step further giving witnesses the option to choose whether or not to issue endorsements based on the implied privacy loss, and uses pseudonyms that change periodically to maintain the privacy of users. In this way, both in APPLAUS [ZC11] and CREPUSCOLO [CCCDP13] the privacy of users is maintained even in scenarios where the LPS is compromised, as locations are associated with pseudonyms that only a trusted CA, that assigns them, knows who owns them.

PASPORT [NSY<sup>+</sup>20] takes a different approach, the prover identifiers are not revealed to witnesses, as this is sent to them encrypted with the verifier public key, as well as witness identifiers (included in the endorsements they issue) are. In this way, only the verifier knows the identity of each user, limiting the privacy exposure of the users.

Furthermore, as peer-based location certification strategies rely on user participation, we need engaging user experience and gamification elements.

## 2.3 User Experience

UX (User eXperience) refers to the experience a user gets when interacting with a product under specific conditions [AT03]. We now explore methods, benefits, and challenges of getting a good UI and UX for end users, as well as how to carry out their evaluation.

### 2.3.1 User Interface

According to Hussain et al. [HHB<sup>+</sup>18] the UI (User Interface) is the part of systems that users interact with and from which they access their features. Thus, the way it is designed makes a difference for the user to find, understand and engage with the features. In other words, designing accessible features contributes to user engagement. However, there are challenges in its design due to the heterogeneity of users (user profile, cognition, disabilities), computing hardware (mobile phone, tablet) and software (OS), Input/Output capabilities (keyboard, touch), among others. To deal with the described heterogeneity, the UI can be adapted, for example, to automatically change the UI color (considering the meanings of the user culture), increase the font size (for visually impaired users or when the user is moving), simplify the UI by hiding or showing widgets (for novice users), among others. Many systems have used a fixed role-based

UI, which consists of presenting a UI tailored to the user roles. However, they are unable to react to contextual changes. To face these challenges the authors proposed a model-based adaptive UI by evaluating UX and context of use. UX is estimated through user feedback, either implicitly by monitoring the user behavior when using the app or by explicitly asking the user for their opinion of the UI.

### 2.3.2 Empirical User Experience Evaluation

The goal of UX is to increase user satisfaction, which is why the evaluation with users is extremely important to extract feedback about the app.

Arhippainen et al. [AT03] clarify how UX can be evaluated in adaptive mobile apps. The authors claim that UX results from the interaction between user, product, context of use, social and cultural factors. Hence, the methods chosen as the most suitable to carry out the evaluation were *interviews* and *observations*, as they allow the construction of a peaceful environment where it is easy to obtain information about the background, expectations, and motivation of the user. Furthermore, the authors reinforce the idea that prototyping is necessary in the preliminary stages of development to obtain user feedback and guide development according to the opinions and needs identified by users.

The authors provide the following advice on user test design:

- *Split Feature Testing*: When there are too many features for users to learn all at once and then be asked about them all, the authors advise splitting testing into sessions where different features are tested so that the user is not overwhelmed with so many features at the same time, allowing them to focus better on each of them;
- *Goal Establishment*: It is important to establish a goal that is expected to be achieved, to know what information needs to be captured to do so efficiently and straightforwardly;
- *Simple Questioning*: The questions must be formulated in a clear and simple way so that they are easily understood, so as not to exaggerate the technicality of the questions and make the user as comfortable as possible;
- *Diversity of Participants*: User testing should cover the widest range of users, so users with different backgrounds should be selected;
- *Observation*: During the interview, to better interpret the user, it is recommended to observe the non-verbal language of the user, however, because this annotation is quite difficult, the authors concluded that it is better to record the interview.

The authors evaluated the following two apps, in which users followed a scenario with the

most relevant use cases and then used the methods mentioned above to obtain user feedback: The first app adapts to the context in which it is inserted (allowing the user to disable the adaptive features), as such, it analyzes the user habits to make better decisions such as changing the sound mode when the user enters a silent place; The second app uses sensors to interact with a map in its interface, examples of which are a compass and an accelerometer, the compass is used to automatically rotate the map based on the user orientation, the accelerometer is used to scroll the map by tilting the device forward, backward and sideways. In addition, the user can calculate the distance to the different places by clicking on them.

### 2.3.3 User Experience in Mobile Augmented Reality

According to Dirin et al. [DL18], AR (Augmented Reality) has become an alternative way to engage users with apps. However, care must be taken not to compromise UX, either by draining the battery or slowing down the performance of the app on the mobile device.

The authors evaluated two case studies of mobile AR apps (both containing an AR animated avatar): a commercial advertisement app and a virtual campus tour app. To assess the usability of these case studies, the *think-aloud method*, in which users are asked to concurrently verbalize their spontaneous thoughts while performing tasks, was applied to understand how the participants think and whether the UI was logically designed. In addition, the time participants took to complete each task was noted to determine the effectiveness of the app.

The user evaluation of these case studies was carried out according to a mixed method approach, encompassing both quantitative and qualitative methods:

- *Quantitative*: Participants were asked to complete a questionnaire with basic personal information and questions rated on a Likert scale [Lik32] to understand what emotions and concerns the participants felt during the experiment;
- *Qualitative*: Afterwards, the participants were interviewed about their experience while using the app, where they were also asked what changes they would make to improve it. The analysis of answers and expressions was performed by applying the coding method in a word processor in which keywords were assigned to the qualitative data excerpts.

The results allowed the authors to conclude that positive emotions were experienced by most users, while negative emotions had much less agreement between them. Furthermore, it revealed that the participants felt involved with AR, as it made the app more interactive. It was also concluded that the avatars caused emotions to the participants, such as relaxation by providing a friendly environment, enthusiasm for the behavior and interaction with the avatar,

and encouragement by helping them and motivating them to explore further and not being afraid of making mistakes or being lost.

Interesting suggestions that participants gave in the context of the virtual campus tour app were that game concepts could be applied, such as missions that could be performed in different places of the visit and background music, arguing that it could make the visit more exciting and relaxing, respectively.

### 2.3.4 User Experience in Location Certification Systems

According to Francisco [Fra21] it is highly important that UX is not overlooked in location certification systems, which can become too security-focused, so that users do not lose interest in them. However, in the past, usability of these systems has not been a top concern.

SurePresence [Fra21] is a location certification system designed with the aim of presenting a seamless UX. *Authentication ceremonies* are security and usability challenges as they take time and may even consist of multiple authentication steps that the user must complete to gain access to the system. One solution is to make users believe that their assets can be attacked and authentication ceremonies can effectively prevent this. Another solution, which was used in this system, is to develop alternative techniques that do not force the user to authenticate. Finally, the user evaluation of this system consisted of the following methods:

- *Quantitative*: Participants were asked to complete a questionnaire, which consisted of user characterization, user experience while using the system and perceived vulnerability of each location certification technique. For this questionnaire, users were asked to rate the statements on 5-point Likert scale [Lik32];
- *Qualitative*: Participants were interviewed about the usefulness of the system, compared to existing ones, and preference for each location certification technique.

One of the ways to engage the user can be through the feelings provoked by the game experiences, since the tourist experience can be adapted to such a context.

## 2.4 Gamification

Gamification has been broadly defined as the use of game elements in non-game related contexts [TF17]. The idea we explore here is to add gamification features to the app to increase user motivation and participation.

According to Hamari et al. [HKS14], gamification can have a psychological effect on users, increasing their motivation and enjoyment in using the app, which can result in behavioral changes, making them more committed to the app.

According to Xu et al. [XWB13], when designing gamification features, it is important that they do not lead to fatigue by themselves. It is necessary to develop meaningful activities that involve gains and losses, to challenge and teach users. Otherwise, if the user does not engage with such features, the so-called *overjustification effect* [TF17] [XWB13] can be triggered, which consists in the user interpreting the addition of rewards for completing an activity as a loss of its value or meaning, or feeling influenced or controlled by it. To engage users, gamification features must encompass intrinsic motivation elements, which concerns the pure pleasure of performing the activity, instead of just extrinsic motivation elements, consisting of getting a better score or being rewarded for it, which, despite encouraging the adherence of users, may not make it lasting if the activities are not really meaningful to them.

#### 2.4.1 Gamification in Tourism

Xu et al. [XWB13] investigated how motivational elements can be used in gamification to have a beneficial impact when used in the tourism domain.

The authors highlighted the following *intrinsic reward* categories:

- *Relatedness*: Allows users to connect and interact with others who share the same interests. A means of communication could be implemented through which users could share their experiences and make recommendations. However, moderators may be needed to prevent users from disclosing false or misleading information;
- *Competence*: Provides users with the feeling of being able to achieve their goals. Means of congratulating users for completing activities could be adopted;
- *Autonomy*: Gives users the opportunity to join the app or opt out at any time, without any penalty.

The authors also highlighted the following potential *benefits* that gamification can pose to the tourism domain:

- *Encourage Engagement*: The pleasure of playing and the desire to keep playing results in user engagement. The rewards that users earn for completing activities in game loops trigger different types of strong emotions, such as excitement, that further involve the user in the gameplay;

- *Enhance Experiences*: Most tourist experiences do not cover intrinsically interactive motivation, social play, challenge, however they can be covered using gameplay elements and thinking;
- *Improve Loyalty*: Gamification can increase user loyalty/engagement by developing interactions not only between the user and the system, but also between different users of the same system, making it unique/nonexchangeable;
- *Increase Brand Awareness*: Gamification can generate opportunities to offer players special promotions.

Next, we show examples of gamification features that have already been implemented and evaluated in other use cases.

### 2.4.2 Gamification of an Engineering Course

Barata et al. [BGJG13] studied the impact of gamification on learning at a graduate level in an engineering course, comparing two versions of the course (original and gamified). Here are some of the features that contributed to student engagement:

- *Experience Points*: Students earn points for academic activities, which provide them with immediate feedback and a sense of reward, motivating them;
- *Levels*: Students are positioned at different levels according to the amount of experience points earned. These levels map directly to the grade students will receive on the course, giving them a sense of progress as they level up;
- *Leaderboard*: A leaderboard was also introduced where students are positioned according to their level and amount of experience points earned, stimulating a competitive spirit that can make them work harder to climb;
- *Challenges*: Students also face alternative challenges, providing different ways to progress, so they can choose the ones they like best, being rewarded with experience points and badges, which are also displayed on the leaderboard. In addition, some challenges are made up of several stages of increasing difficulty, depicting progress, and students can decide whether they want to complete it fully or not.

### 2.4.3 Gamification-Based Incentive Mechanism

Typically, the incentives used to encourage people to participate in activities are based on monetary rewards. But gamification could represent an alternative way to make users feel satisfied, while reducing the amount of monetary rewards that are otherwise awarded. Ueyama

et al. [UTAY14] proposed an incentive mechanism in a participatory sensing system, which aims to minimize the value of monetary rewards based on the following schemes:

- *Level*: Based on the points users earn from activities, they are categorized into different levels. Users who are at higher levels earn more points per activity than others. This way, users are encouraged to perform more activities to level up and earn more points;
- *Badge*: When the user fulfills a certain condition, they receive a badge. The badges obtained must be visible to dazzle other users and motivate them to complete the activities necessary to acquire badges;
- *Ranking*: Users are placed in a rank accessible to everyone being urged to work harder to climb.

In this system, the points earned can be exchanged for cash at any time, however, due to the way the accumulation of points works, when the user makes the exchange and is therefore left with less points, it means they will not earn as many points for completing activities as they would have if they had not traded them, posing a trade-off for them to decide whether they intend to maximize the accumulation of points or exchange them for cash. Associated with this trade-off, the system also has a heuristic algorithm to decide to which users should be sent reward activities for them to complete, based on their reward points.

The evaluation of this system was carried out through questionnaires aimed to determine the effectiveness in motivating users to participate in the activities. The level and ranking schemes were considered by users as the most effective of the implemented gamification schemes.

#### **2.4.4 Gamification of a Location-Based Mobile Application**

Thiel et al. [TF17] investigated the effects that gamification can have on location-based public participation and to what extent motivation influences participation levels. Based on the evaluation of other apps, the authors claim that users were strongly motivated by: community membership, having fun, and learning. Another finding was that apps in which many gamification features are integrated can become complex, raising the question of how many features are too many.

The authors gamified an app prototype to compare it with its non-gamified version. The implemented features were the following: social interaction; emoji reactions so that users can express their mood towards a specific topic; user profile that maintains their activity within the app; scoreboard to foster competition; and quests to get information about specific topics.

This study was evaluated through a user survey on aspects of personal relevance from different categories that map to either intrinsic (social, pleasure and learning) or extrinsic (reputation, institutional and personal) motivation. The evaluation sought to find out what type of motivation most affected participation levels. However, the authors were unable to identify a direct influence, although they concluded that both had a positive effect.

## 2.5 Summary

As seen, all location certification strategies developed in the CROSS City prototype are infrastructure-based, as such they have inherent restrictions (e.g. operation confined to spaces with required infrastructure deployed; trivial transfer of evidence between users; temporal evidence requires the use of modified infrastructure). To address it, we will integrate a peer-based strategy. In peer-based location certification systems [FP18, TCB10, ZC11, CCCDP13], mechanisms are designed to make endorsements nontransferable and unforgeable, and to ensure user location privacy and resilience against collusion between users. Table 2.1 illustrates which properties are guaranteed in these systems and which are desirable for CROSS City to secure.

UX and gamification can boost user engagement, motivation and participation, which is a huge benefit that we are definitely interested in exploring in the CROSS City app that was not its prototype. Now we are even more motivated to do so as the peer-based location certification strategy relies on user participation. To improve UX, it was seen that it is important to design a UI that is adaptable to the mobile device and the user who uses it [HHB<sup>+</sup>18]. In addition, mobile phone sensors can be used to allow the app to adapt and leverage them to improve user interaction [AT03]. The evaluation of UX was carried out in most cases both quantitatively with questionnaires and qualitatively with interviews [DL18, Fra21].

Gamification can bring potential benefits to the domain of tourism. In the gamification use cases listed [BGJG13, UTAY14, TF17], several features were pointed out. In this regard, it should not be overlooked that, in addition to designing gamification features that provide extrinsic motivation, it is also necessary to design features that provide intrinsic motivation, as these are the motivational elements that will make the system gain meaning for users [XWB13]. For the evaluation of the gamification features in most use cases, questionnaires were used to determine how they affected user motivation.



Table 2.1: Properties of peer-based location certification systems.

	<i>LINK</i>	<i>APPLAUS</i>	<i>CREPUSCOLO</i>	<i>PASPORT</i>	<i>CROSS</i>
Works properly without surrounding infrastructure	✓	✓	✗	✓	✓
Works properly without witnesses	✓	✗	✗	✗	✓
Resistant to eavesdropping on messages	✗	✓	✓	✓	✓
Resistant to tampering with messages	✗	✓	✓	✓	✓
Resistant to replay attacks	✓	✓	✓	✓	✓
Resilient against prover-witness collusion attacks	✓	✓	✓	✓	✓
Resilient against prover-prover collusion attacks	✗	✗	✓	✓	✓
Unwanted server access does not reveal user location history	✗	✓	✓	<i>N/A</i>	✗
Number of guaranteed properties	4/8	6/8	6/8	6/8	7/8



# Chapter 3

## Implementation Preliminaries

In this Chapter we present the preliminaries for the implementation of this work. We begin by presenting the extended architecture of CROSS City. Next, we present the details of the CROSS City re-implementation, to create the base system for receiving the new contributions.

### 3.1 Architecture Overview

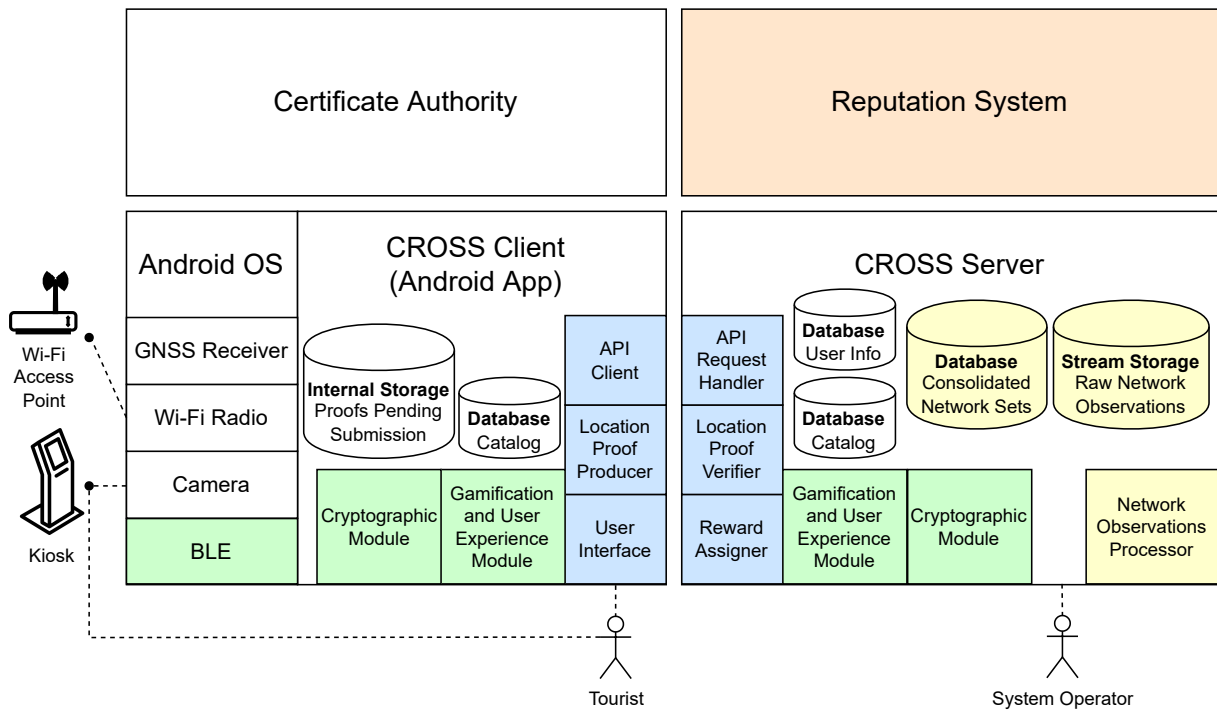


Figure 3.1: Extended architecture overview of CROSS City.

The points of interest of a city can be public or private. We assume that each is represented by a point-of-interest representative. We also assume that the city has a tourist office that collects point-of-interest information. The system operator is the entity that manages the overall CROSS City system, they leverage point-of-interest information to produce the CROSS City catalog, which contains the name, description and location of each point of interest and organizes them into routes.

Figure 3.1 illustrates the extended architecture overview of CROSS City:

- The components (listed below) in ■ (green) and ■ (blue) were, respectively, added and modified by this work;
- The reputation system in ■ (orange) was added by the work of Figueiredo [MP22b];
- The server components in ■ (yellow) were added by the work of Vicente [MP22a].

For each new (green) component that was added by this work, its purpose is:

- *Gamification and User Experience Module*: Responsible for the gamification and UX functionality logic incorporated into the app;
- *Cryptographic Module*: Responsible for encrypting and authenticating messages so that the entities can communicate securely without compromising the confidentiality or integrity of the messages exchanged;
- *BLE*: The communication protocol over which users endorse the claims of their peers.

For each existing (blue) component that was modified by this work, its purpose is:

- *User Interface*: With the integration of the new functionality, the existing UI underwent modifications and extensions;
- *API Client and Request Handler*: It was necessary to rebuild and add new API endpoints, to support the new features and the new peer-based strategy;
- *Reward Assigner*: With the integration of gamification features, the rewards and the method of assigning them to users underwent changes;
- *Location Proof Producer and Verifier*: The new peer-based strategy added a new method of producing location proofs, from acquiring peer endorsements on the client and validating them on the server.

## 3.2 CROSS City Re-Implementation

In this Section we present the re-implementation details of each component of the CROSS City system, i.e. server, contract and mobile application. At the beginning of each component we




explain the reasoning behind the re-implementation and then the adopted technologies, data model and main changes.

### 3.2.1 Server

We re-implemented the CROSS City server [MCP20] as, in our common perception, the original code was poorly maintained and documented.

**Server development technology** The programming language chosen to develop the server was Java (seeking to homogenize the server development language across the SureThing projects) using Maven<sup>1</sup> as the build automation and dependency management tool. We use the Jersey<sup>2</sup> framework as the implementation reference of Jakarta<sup>3</sup>, which simplifies the development of RESTful web services (e.g. by providing additional Java annotations) and allows us to abstract from the low-level details inherent to client-server communication. As for the server application, we use GlassFish<sup>4</sup> with Grizzly<sup>5</sup> as the HTTP listener for incoming requests designed to build robust and scalable servers.

**Server database** PostgreSQL<sup>6</sup> was maintained as the database management system as the relational database schema fits our data schema. For database accesses to be scalable, we use HikariCP<sup>7</sup>, which is a lightweight, fast JDBC connection pool framework, achievable by maintaining an in-memory cache of database connections so they can be reused in subsequent requests.

**Database schema** We now present the database schema illustrated in Figure 3.2, which is an extension of the Maia et al. [MCP20] one. The tables and fields in  (green) are related to the gamification features presented in Section 4.4; The tables in  (purple) are related to visits validation, in particular, the *PeerTestimony* was introduced to maintain peer endorsements and the *UserCryptoIdentity* to verify their authenticity, explained in more detail in Section 4.3.2.2; The tables in  (blue) are those intended to maintain the system catalog, they have been unchanged or slightly modified according to new needs (as in the case of points of interest and routes to which descriptions and an image have been added).

---

<sup>1</sup><https://maven.apache.org/>

<sup>2</sup><https://eclipse-ee4j.github.io/jersey/>

<sup>3</sup><https://jakarta.ee/specifications/restful-ws/>

<sup>4</sup><https://javaee.github.io/glassfish/>

<sup>5</sup><https://javaee.github.io/grizzly/>

<sup>6</sup><https://www.postgresql.org/>

<sup>7</sup><https://github.com/brettwooldridge/HikariCP>

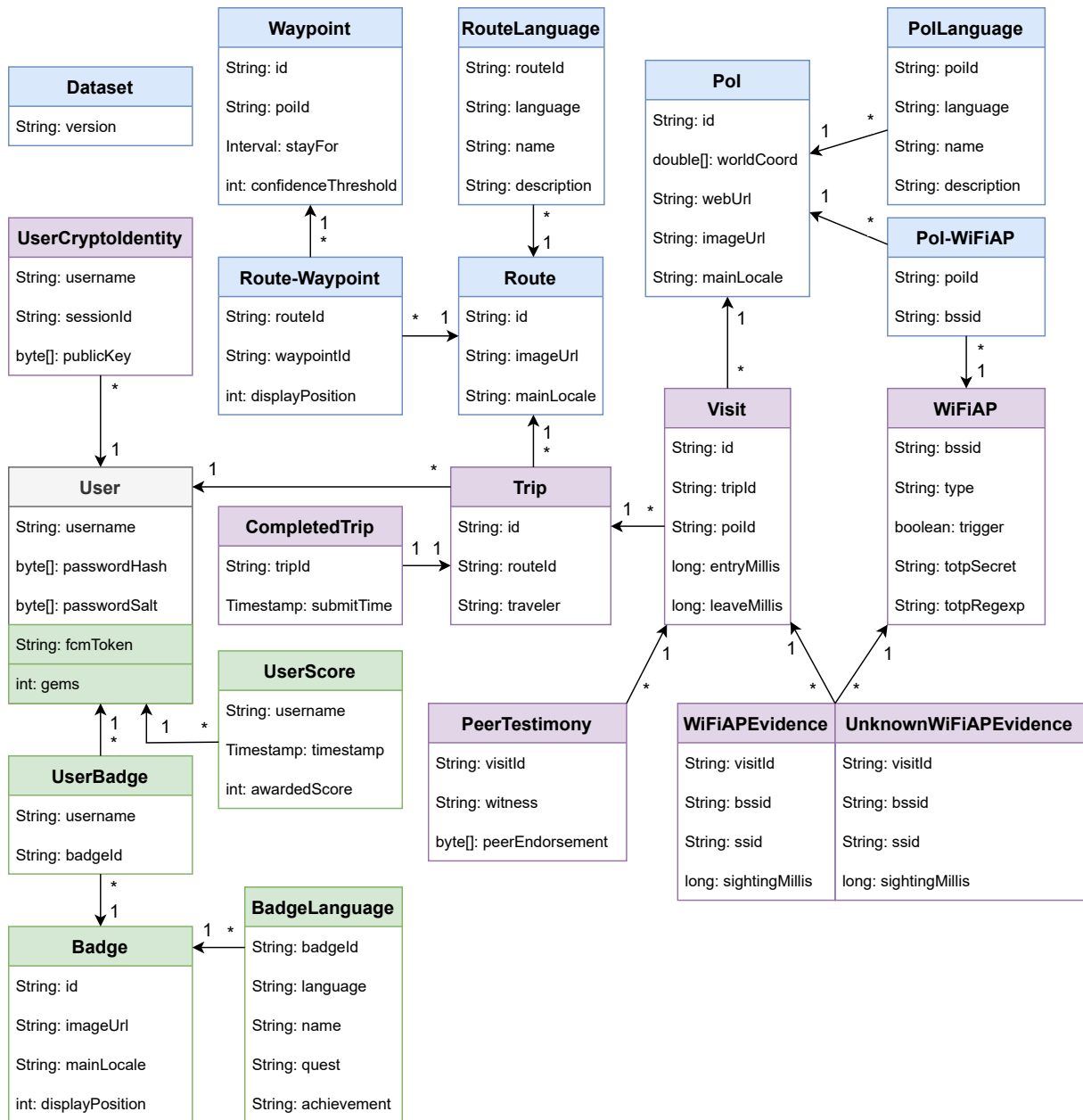


Figure 3.2: Server database entities and relationships between them.

**Main changes** The main changes made during the server re-implementation that modify the way it works were as follows:

- We now allow visits to points of interest to be submitted individually instead of grouped by route, which allows users to get real-time server feedback;
- We removed the order in which points of interest must be visited by route (giving them just a suggestion, displaying them in a certain ordered list), this gives users more flexibility according to their position framed within the route;
- Upon user authentication, a JWT is now issued to be sent in the header of subsequent

client requests as an (efficient and standard) authorization method (necessary, as REST is a stateless protocol, where there is no concept of connections).

### 3.2.2 Contract

In the original CROSS City server [MCP20] the client-server messaging specification was defined in Go structures, which is not maintainable. As such, we have reconstructed the contract in protobuf<sup>8</sup> that allows us to serialize our typed data across different platforms and languages (while also ensuring compliance across SureThing projects message typification).

**Lite contract** The gRPC dependencies that were being used to compile the contract protobuf messages for the server to use are not supported by Android. So we created a new module that compiles the contract for Android using the lite dependencies that gRPC provides.

**Documentation** To document the API that the server exposes, we use the gnostic protobuf compiler plugin<sup>9</sup> that automatically generates an OpenAPI<sup>10</sup> description for a RESTful API (i.e. documentation about the data and API specification) directly from a protobuf contract.

### 3.2.3 Mobile Application

This mobile application is an extended re-implementation of the CROSS City app prototype previously developed by Maia et al. [MCP20]. This new version of the app was developed from scratch, because: the project dependencies were already quite outdated; the server communication API (introduced in 3.2.2) has been completely rebuilt with a new messaging specification<sup>8</sup>; the app logic could be simplified; and finally, for these reasons and the need to understand the source code in detail, we decided to perform an incremental development, studying the existing app source code as we were developing it.

**Communication with the server** Firstly, we developed the client API for communication with the server. We use Retrofit<sup>11</sup> as the HTTP client because it is type-safe, efficient, it possesses a simple syntax and supports a protobuf converter<sup>12</sup> to handle protobuf message serialization under the hood, which is highly convenient for our use case as our API contract is

---

<sup>8</sup><https://developers.google.com/protocol-buffers>




<sup>9</sup><https://github.com/google/gnostic/tree/main/cmd/protoc-gen-openapi>

<sup>10</sup><https://swagger.io/specification/>

<sup>11</sup><https://github.com/square/retrofit>

<sup>12</sup><https://github.com/square/retrofit/tree/master/retrofit-converters/protobuf>

specified in protobuf. To ensure the confidentiality and authenticity of client-server messages, we configured TLS with the certificate from the private CA we setup (intended to be used across all SureThing projects) used to issue the RSA key pair that servers use to establish HTTPS connections. To authorize client requests, the user saves the JWT (issued by the server upon user authentication) to embed it in the header of requests where authorization is required, automatically/transparently renewing it when it expires.

**Data persistence** To persist data on the mobile device, we use the Room <sup>13</sup> database object mapping library, which provides an API that allows to conveniently persist entity classes that can be used as domain objects, providing an abstraction layer on top of SQLite <sup>14</sup>, which is built into Android, is cost-effective, efficient, and reliable. The modeling adopted for our data is illustrated in Figure 3.3 (already designed to support our contributions to the system). The tables in  (green) are related to the gamification features presented in Section 4.4; The tables in  (purple) contain the visits validation data. The collected location evidence must be persisted so that, if there is no Internet access, it can be submitted to the server in a later interaction; The tables in  (blue) are kept to store the system catalog both to support the offline use of the app, but also to avoid wasting network resources by downloading it every time the app is launched, for that, the catalog version is also kept in the shared preferences, so that the catalog update check is performed by comparing only the kept version value with the latest one instead of their contents.

As illustrated in Figure 3.3 the images of routes and points of interest are listed from downloadable URLs, to load them we use Glide <sup>15</sup> an image loading framework that downloads images and automatically manages memory and disk caching to support the offline use of the app.

Another place to persist key-value data is in the shared preferences: configuration data (such as user-unique values or app settings) is stored there in plain text; secret data (such as user credentials, so that when the app is launched, after successful authentication, the user does not need to manually authenticate again) is stored there encrypted.

---

<sup>13</sup><https://developer.android.com/reference/androidx/room/package-summary>

<sup>14</sup><https://www.sqlite.org/index.html>

<sup>15</sup><https://github.com/bumptech/glide>



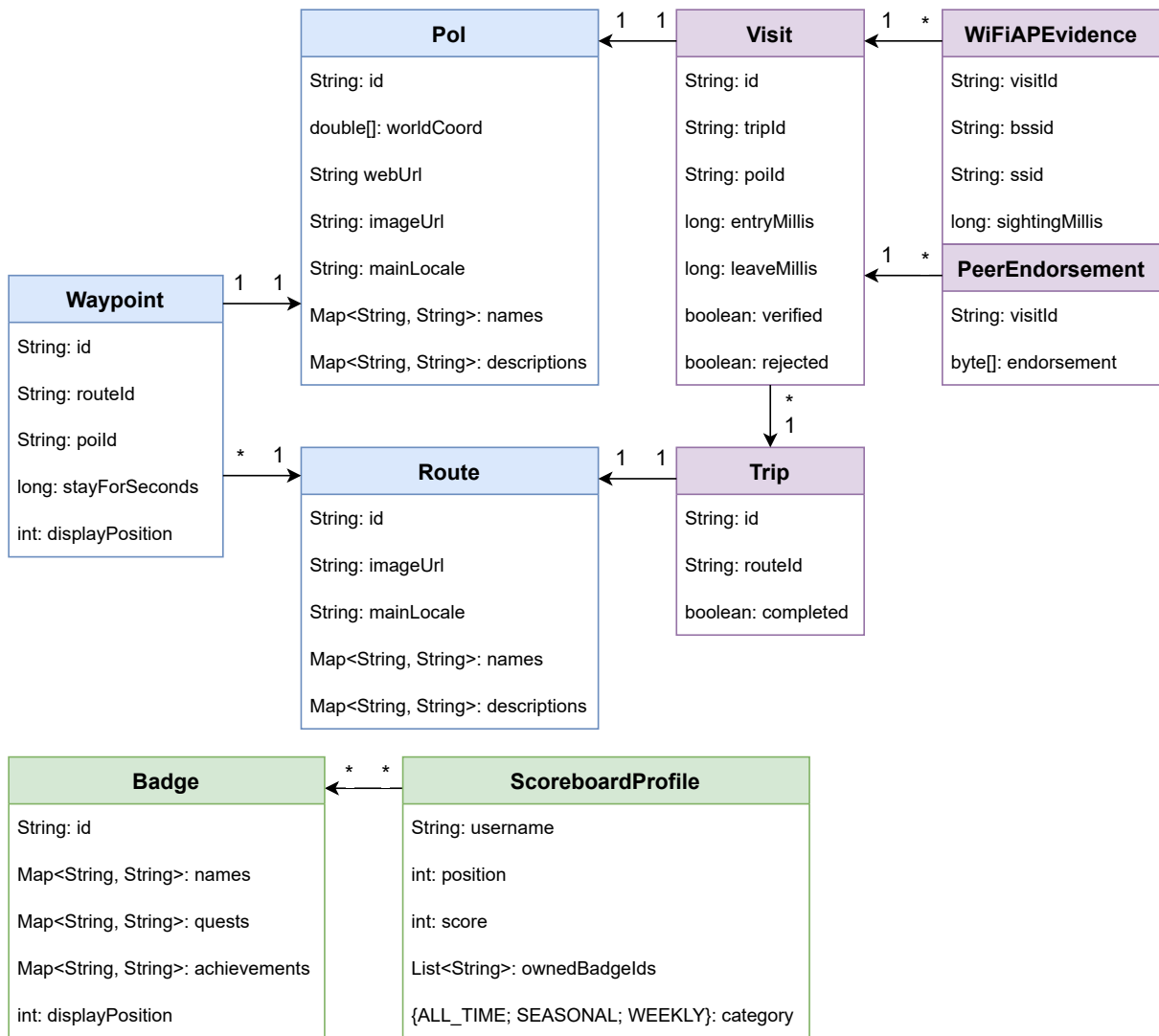


Figure 3.3: Mobile application database entities and relationships between them.

**Extended functionality** With the rebuild of the app, in addition to all the previous functionality developed in the app prototype, it is important to mention the new ones incorporated:

- When the user gets authenticated, we will now fetch the trips and visits the user has taken so far to support cross-device usage or app uninstalls. This fetching, like catalog synchronization, or queued visits submission (when Internet access is re-established), was implemented using `WorkManager`<sup>16</sup>, which is the Google recommended API for background processing;
- We divided the routes into different sections within the app: traveled and untraveled routes, so that the user can remember the ones already traveled and explore in a concise way the ones that are still to be traveled; and initiated routes, so users can see their

<sup>16</sup><https://developer.android.com/reference/androidx/work/WorkManager>

- progress on each and the status of each visit (verified, rejected, or queued due to the absence of Internet access). We have also added images and descriptions to the different routes and points of interest;
- For the user convenience we added a button where the user can get directions to the point of interest, which redirects them to Google Maps <sup>17</sup> with the geocoordinates of that point of interest as the destination, using Google Directions API <sup>18</sup>;
  - As we have developed the support of submitting single visits to the server, it is now possible to provide immediate feedback to the user at the end of the visit, as long as Internet access is established. Therefore, when the visit gets rejected, we added a rejection alert dialog to ask if the user is sure they want to end the visit anyway, or extend it a little further to acquire the location evidence required for its validation. To provide user feedback, we have also added information on the end-of-visit screen where the user can see the number of Wi-Fi scans performed and the number of location evidences (Wi-Fi APs) collected;
  - We incorporated the Wi-Fi scanning cycle into a foreground service <sup>19</sup> (background services <sup>20</sup> have a limited duration of 10 minutes, which is not suitable for our use case), so scans do not stop even if the app is not in the foreground, although due to Wi-Fi scan throttling in newer versions of Android, it is only allowed to perform 1 background scan in a 30-minute period as opposed to the 4 foreground scans allowed in a 2-minute period, but this way, when the app comes back to the foreground, the scans continue automatically;
  - Finally, as the main objective of this work is to incorporate a new peer-based strategy into the app, we added a settings entry that allows the user to configure which location certification scheme they want to use.

**Release of the first version of the app** With the app logo and theme updated to give the app a fresh look and all the features present in the app prototype developed (with optimized source code, updated dependencies, pleasant UI, and ready to integrate subsequent functionality). To ensure that everything was implemented correctly, we set up a route through the Taguspark campus of Instituto Superior Técnico. With the list of access points present at each visit point in the server database and the server [MP22a] deployed on Google Cloud <sup>21</sup>, the user is ready to go through the route. We asked a couple of our colleagues to travel this route to attest if they were able to successfully validate the route, and everything went as expected.

---

<sup>17</sup><https://maps.google.com/>

<sup>18</sup><https://developers.google.com/maps/documentation/directions/get-directions>

<sup>19</sup><https://developer.android.com/guide/components/foreground-services>

<sup>20</sup><https://developer.android.com/reference/androidx/work/WorkManager>

<sup>21</sup><https://cloud.google.com/gcp>

### **3.3 Summary**

We completely re-implemented the CROSS City prototype system, including server, contract and mobile application, to update the project, optimize the code base and offer an improved UI, extending it with some additional functionality. With the CROSS City system ready to receive our contributions, let us now dive right into it.



## Chapter 4

# Implementation

In this work, we deliver a new version of CROSS City, where we integrated the new peer-based collusion resilient location certification strategy – named *P2P Witnessing* from here onwards – and gamification elements to further engage and encourage users to use it.

In this Chapter, we first present the assumptions and the solution guidelines of this work. Next, we present the implementation of the P2P Witnessing strategy, along with the peer endorsement acquisition and validation protocols. Finally, we present the gamification elements incorporated into the app.

### 4.1 Assumptions

The P2P Witnessing strategy requires the following assumptions:

- The users are willing to share a small portion of their mobile device power resources to endorse the claims of their peers;
- The user mobile device supports BLE to execute the peer endorsement acquisition protocol;
- The user mobile device will eventually have access to the Internet to validate the user visits.

### 4.2 Guidelines

In this work we are introducing the P2P Witnessing strategy, which must be dependable, collusion-resilient and timely. As such, we defined guidelines for the peer endorsement validation (**G1-3**) and for their acquisition (**G4-6**):

- G1** The strategy must only successfully validate the visit if a considerable number of witnesses endorse it (the better well-behaved track record they have, the less it should be needed);
- G2** The strategy must be resilient against systematic prover-witness collusion;
- G3** The peer endorsement validation protocol must provide responsive user feedback;
- G4** The strategy must be resilient against prover-prover collusion (i.e. the witness endorsement issuance acceptance rate would ideally be close to 0% for colluding provers and 100% for honest provers);
- G5** The peer endorsement acquisition protocol must not require users to be in close proximity for an extended period of time;
- G6** The strategy adoption impact on the mobile device power resources must not impact the user experience.

In addition, as the P2P Witnessing strategy relies on user participation, to ensure its proper functioning, we defined these three additional guidelines:

- G7** The app must make users want to adopt the P2P Witnessing strategy;
- G8** The app must encourage users to further use it;
- G9** The app must work on unmodified smartphones running the most popular OS to reach a large number of users.

## 4.3 P2P Witnessing Strategy

The P2P Witnessing strategy aims to solve the problem of the CROSS City app prototype not working in places where there is a lack of infrastructure around it, and address the security flaws of the existing Wi-Fi-based strategies that allow transfer of location evidence between users, as explained in Section 2.1.

We begin by presenting the threat model. Then, we present the protocols for acquiring and validating peer endorsements. Finally, we explain how the protocols presented mitigate the threats we focus on.

### 4.3.1 Threat Model

We now present the threat model we took into account when designing this strategy, to clarify the attacks we take into account in the scope of the solution.

We assume that a malicious user  $U_\alpha$  **cannot**:

- Share private content (such as private keys and secret numbers), which would allow third parties to deceive witnesses by impersonating  $U_\alpha$ ;
- Compromise the server and gain access to the data it stores;
- Find out the server private key.

On the other hand, we assume that a malicious user  $U_\alpha$  **can**:

- T1** Eavesdrop and tamper with sensitive messages;
- T2** Submit endorsements issued on behalf of other users (e.g. captured by eavesdropping messages) for  $U_\alpha$ 's own benefit;
- T3** Submit endorsements issued in the past to try to certify the current location of  $U_\alpha$ ;
- T4** Systematically collude with witnesses that endorse  $U_\alpha$ 's false claim;
- T5** Collude with another prover, causing honest witnesses to endorse  $U_\alpha$ 's false claim;
- T6** Try to track app users eavesdropping the messages they exchange.

### 4.3.2 Peer Endorsement Acquisition

In this Section, we address the peer endorsement acquisition implementation into the CROSS City app. As such, we start by defining which short-range wireless technology to use for communication between users, then we introduce the acquisition protocol, and finally, we delve deeper into the DB protocol.

#### 4.3.2.1 Short-Range Wireless Technology

The short-range wireless technology chosen, over which users endorse the claims of their peers, was BLE (Bluetooth Low Energy) for the following reasons:

- *Low power consumption*: The app is designed to be used while the user is traveling around the city, therefore without the possibility of charging their mobile device;
- *Connection speed*: Users may only be within range of each other for a short period of time (a few seconds);
- *Ability to enable device discoverability without user interaction*: Other short-range wireless technologies, such as classic Bluetooth, require the user to manually agree to make the device discoverable for a certain time period, which can represent a deficit for the user experience;
- *Ability to establish a connection without having to pair devices*: Other short-range wireless technologies, such as classic Bluetooth and Wi-Fi Direct, require pairing which, in turn,

requires the user to manually accept such pairing with new mobile devices, requiring user interaction, which can overwhelm the user during a visit;

- *High throughput not required*: The messages exchanged in the acquisition protocol are quite small, not demanding much throughput (main advantage of classic Bluetooth and Wi-Fi Direct over BLE, but not an advantage for our use case);
- *Extensibility to limited devices*: Allows the future deployment of a peripheral in wearable beacons with limited power resources to act as witness and endorse the user location.

Before diving into the actual implementation, let us now introduce fundamental concepts to programming in the BLE interface. Mobile devices can play two distinct roles: *Central* is the one that discovers peripheral devices and sends them a connection request; *Peripheral* is the one that advertises itself, making the device discoverable to central devices, and accepts their connection request.

To exchange messages between devices via BLE, the most straightforward method we identified and therefore adopted is the following: *Central*  $\rightarrow$  *Peripheral*: The central writes to a characteristic (field whose value is stored on the peripheral) and the peripheral listens for changes in this value; *Peripheral*  $\rightarrow$  *Central*: The peripheral writes to a characteristic and the central is notified of the change in that value (for this it is necessary to subscribe in advance to this characteristic change notifications).

It also important to take into account that the maximum message payload size is 512 bytes, which is the  $\min(MTU, \maxCharacteristicLength) = 512$  bytes, being that (according to Google) the maximum requestable MTU on Android is 517 bytes and the maximum characteristic length is 512 bytes.

With the short-range wireless technology chosen and the research carried out on it, we have the knowledge required to dive into the implementation of the acquisition protocol.

#### 4.3.2.2 Peer Endorsement Acquisition Protocol

Firstly, users need to be able to sign claims and endorsements, so we first explain how users get a cryptographic identity within the system. Next, we explain the details of BLE communication between users. Finally, we delve into the steps of the acquisition protocol without, for the moment, delving too deeply into the DB protocol included in it.

**Cryptographic key distribution** Users need an asymmetric key pair, whose public key is known to the server, **so it can validate their claims and endorsements signatures**. To



ensure that the private key is in the exclusive possession of the user, we decided to generate the key pair in the mobile app when the user logs in. This way, the generated key pair is RSA-2048 bits stored in the Android keystore. The user public key is shared with the server at user authentication time, as it is embedded in the authentication request, as this data and all other data exchanged with the server are sent via HTTPS, it is ensured that the user who is sending the public key is in fact in possession of the account password. The HTTPS cryptographic keys were signed by a CA, which we created (intended to be used by the SureThing systems), whose certificate was manually embedded within the mobile application bundle. We allow the submission of a public key in all authentication requests, regardless of whether it represents the registration of a new account, to allow the user to connect on different mobile devices. In this same context, on the server we do not just store a public key associated with a user, but a set of public keys, as each of them can be associated with the same account on different mobile devices. Therefore, so that the server does not have to try all the public keys associated with an account to validate a signature, we assign each key pair a session identifier (the session period being from the time the user logs in until the user logs out), and the user must embed in the signed messages the session identifier associated to the key pair used to sign it. Furthermore, in response to the authentication request, the server sends its public key so that users can encrypt data visible only to the server, thus safeguarding their identity, the server public key is also stored in the Android keystore on the mobile device, to be used in cases where there is no Internet access.

**BLE communication between two users** When the user starts the visit, a foreground service is launched, responsible for the components that make up this strategy. The foreground service starts both: advertising itself to nearby devices and periodically scanning (with duration and period of 5 seconds) for them. To ensure that the devices they connect to are app users, the advertising data contains the identifier of the service it provides, which is unique to our protocol. When a device is scanned, if an instance of the protocol is already running, that device enters a queue (this is done so that there are not multiple protocol running instances, otherwise it can affect its performance and latency is a critical factor for its success), when the device turn comes, a connection request is sent to it. When the scanned device (witness) accepts the connection request and the connection is established, the client device (prover) requests a high-priority connection to keep the latency of subsequent messages to a minimum and negotiates an MTU of 515 bytes (the maximum message payload size of our protocol is the endorsement, which is a ciphered block with the server public key RSA-4096, so 512 bytes, the header size of messages

exchanged via BLE is 3 bytes, so we require an MTU of at least  $512 + 3 = 515$  bytes, that is fine because Android devices allow an MTU of up to 517 bytes). As soon as the connection is set up, the acquisition protocol execution itself begins. When the user collects or issues endorsements, this information is captured to be displayed on the end-of-visit screen as feedback to the user. To handle errors that might occur during connection establishment, we retry it under a linear backoff policy (i.e. the reconnect timeout increases linearly). When the protocol execution ends, that is, the claim is rejected or the endorsement is issued, the client device ends the connection. When the user ends the visit, this foreground service is destroyed after canceling the periodic scanning, stopping advertising itself, and correctly terminating all current connections.

Let us now go through the steps of the acquisition protocol. Let  $E_{entity}$  be the encryption function with the entity public key using the *RSA/ECB/PKCS1Padding* algorithm,  $S_{entity}$  the signature function with the entity private key using the *SHA256withRSA* algorithm, and  $m_i||m_j$  the concatenation of  $m_i$  with  $m_j$ .

1. **Prover:** Upon starting the visit, the prover generates two  $n$ -bit random numbers  $a$  and  $b$  (which should not be reusable);  $n$  corresponds to the number of challenges that the protocol is composed of (value presented below);
2. **Prover:** Upon crossing paths with a witness, the prover generates the claim  $LC = a||b||ID_{prover}||SessionID_{prover}||PoI||Timestamp$  and sends its signature  $e = S_{prover}(LC)$  to the witness;
3. **Witness:** Upon receiving the claim signature, if the witness has not yet endorsed the prover <sup>1</sup> or the prover has not yet exhausted their attempts, the witness generates an  $n$ -bit random number  $h$  and sends it to the prover;
4. **Prover:** Upon receiving  $h$ , the prover computes  $z = b \oplus h$  and responds to the witness with an acknowledgement message;
5. **Witness:** Upon receiving the acknowledgment, the witness generates an  $n$ -bit random number  $c$  and begins the DB protocol: For each challenge bit  $c_i$ :
  - (a) **Witness:** The witness sends  $c_i$  to the prover, starting a timer immediately thereafter to time the response;
  - (b) **Prover:** Upon receiving  $c_i$ , the prover computes the response bit  $r_i = a_i$  if  $c_i$  is 0, otherwise  $r_i = z_i$ , and sends  $r_i$  to the witness in response to the challenge;
  - (c) **Witness:** Upon receiving  $r_i$ , the witness validates the response time to the challenge (validation which is explained in detail below in the DB protocol presentation), if it fails the claim is rejected.

---

<sup>1</sup>The MAC address is used to distinguish provers as the device name is not disclosed to avoid user tracking.

6. **Witness:** Upon successful completion of the DB protocol, the witness generates  $m = r||c||h||e||ID_{witness}||SessionID_{witness}||PoI||Timestamp$  and sends the endorsement  $LE = E_{server}(m)||S_{witness}(m)$  to the prover.

**Message format** All messages exchanged between users in these steps are protobuf messages to structure and type their data, these messages are compactly serialized and efficiently deserialized by the receiver.

To summarize, the prover-witness message flow of the acquisition protocol is illustrated in Figure 4.1.

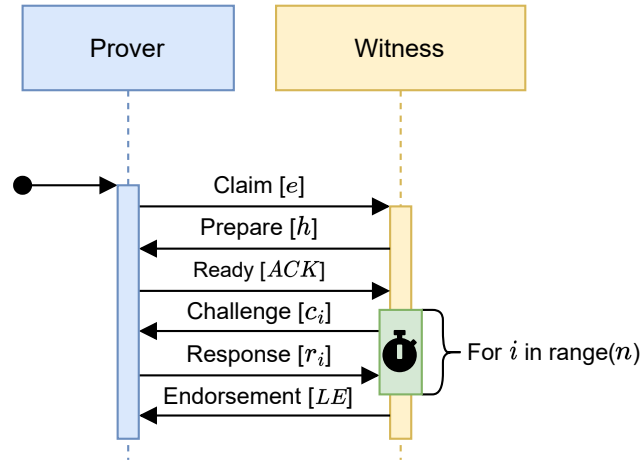


Figure 4.1: Prover-witness message flow

This protocol was inspired by the P-TREAD protocol introduced in PASPORT [NSY<sup>+</sup>20], in which the verifier is the server, the main change made besides the validation of the response times of the challenges explained in detail below, is that there is no first contact with the server to generate a claim identifier. This step was necessary for the PASPORT witness selection phase, which aims to reduce the effectiveness of prover-witness collusion, with only a few witnesses of the server choice able to issue endorsements to the prover in question. However this solution is not suitable for our use case as users are not required to have Internet access during the visit and as such the server is unaware of which witnesses are at the same location as the prover. Instead, to be resilient to prover-witness collusion we integrated the reputation system [MP22b] and the witness decay mechanism explained further below.

### 4.3.2.3 Distance-Bounding Protocol

Inspired by P-TREAD introduced in PASPORT [NSY<sup>+</sup>20], we initially planned to define the maximum acceptable challenge response time  $RTT_{max} = \frac{2D}{C} + t_0$  where:  $D$  is the maximum distance that can be found between the prover and the witness;  $C$  is the speed of light; and finally,  $t_0$  is the computation overhead of the challenge response bit. However, when implementing it, we found that this threshold did not hold, the response times we were measuring (between about 20 and 40 milliseconds) were  $\gg$  than  $RTT_{max} \approx 1$  millisecond (considering  $D = 10$  meters and  $t_0 \approx 1$  millisecond), the reason for this is that the propagation time represented in the  $RTT_{max}$  equation by  $\frac{2D}{C}$  is on the order of nanoseconds, but unfortunately this does not take into account factors such as: the BLE connection interval (which delays packet propagation by a few milliseconds even when requesting a high-priority connection <sup>2</sup>), the volatility of the Android system running other processes, or the network congestion. These problems are due to the fact that our DB protocol is fully implemented at the application layer, where we do not have full control over these factors, but we also cannot descend from this layer because we want our app to be usable on any unmodified Android device.

Faced with this problem, we tried to work around it by experimenting with other short-range wireless technologies, to check if this degree of latency in sending packets was general, or unique to BLE. We implemented the protocol in classic Bluetooth <sup>3</sup>, Wi-Fi Direct <sup>4</sup>, and Nearby Connections API <sup>5</sup> (this interface uses a combination of BLE, classic Bluetooth, and Wi-Fi Direct). The implementations on top of classic Bluetooth and Nearby Connections API resulted in higher challenge response times than those achieved with the initial implementation of BLE with the high-priority connection. The only implementation that seemed to give better results beforehand ( $\approx 3$  milliseconds) was the one on top of Wi-Fi Direct, however, the solution did not scale, when we did exhaustive tests where we placed two devices exchanging endorsements every 5 seconds, the response times increased greatly over time, reaching times longer than those obtained with the implementation of BLE. In addition, it forced the users to accept pairing with other devices to exchange endorsements, connection establishment errors were frequent and more battery was drained. As none of the alternative technologies solved this problem, we decided to stick with BLE because, as explained in Section 4.3.2.1, it is the technology that best suits our use case.

---

<sup>2</sup>[https://developer.android.com/reference/android/bluetooth/BluetoothGatt#CONNECTION\\_PRIORITY\\_HIGH](https://developer.android.com/reference/android/bluetooth/BluetoothGatt#CONNECTION_PRIORITY_HIGH)

<sup>3</sup><https://developer.android.com/guide/topics/connectivity/bluetooth>

<sup>4</sup><https://developer.android.com/guide/topics/connectivity/wifip2p>

<sup>5</sup><https://developers.google.com/nearby/connections/overview>

Given the impossibility of using the P-TREAD  $R_{TT_{max}}$  equation for our use case, we decided to empirically define our own challenge response time threshold. The value we determined was  $T_{max} = 34,5$  milliseconds in our evaluation 5.1.2.1 in order to keep the DB protocol execution success rate as low as possible for colluding users and as high as possible for honest users. Furthermore, to provide robustness to the DB protocol against observed response time fluctuation (induced by runtime volatility) without degrading its resilience against colluding provers, we designed the following refinements:

- *Number of challenges definition:* We set the number of challenges that the protocol is composed of to 64, as this seemed to be enough to get an average that would mitigate the response time spikes induced by the Android runtime volatility without degrading its performance.
- *Noise reduction:* To eliminate underlying implicit noise, we do not take into account the 10% worst response times recorded.
- *Claim rejection:* Instead of rejecting a claim as soon as we get a longer-than-expected response time, we sum up the fastest 90% and only if that sum is greater than  $T_{max} \times \text{numberOfChallenges} \times 90\%$  does the witness reject the claim. In this way, the claim is only rejected if the average response time is or will be greater than  $T_{max}$ .
- *Attempts:* Each witness allows each prover 3 protocol execution attempts, in case of false negatives in claim rejection, this flexibility allows us to be more demanding in defining the  $T_{max}$  value in order to reach greater resilience against colluding provers.

### 4.3.3 Peer Endorsement Validation

When CROSS users end the visits, they submit all the location evidence collected during the visit to the server. In this Section we present the peer endorsement validation process, witness decay mechanism and visit confidence calculation.

#### 4.3.3.1 Peer Endorsement Validation Process

Attached to the endorsement, the prover submits the claim to which the endorsement is associated, we recall that the claim is generated by the prover and the endorsement by the witness.

**Claim validation** The first step to validate the evidence is to validate the claim, in this way if any of the following conditions are met the claim is invalid:

1. The prover named in the claim is not the visit validation requester (extracted from the

- JWT sent along with the request). *Rationale*: The prover is trying to use evidence issued on behalf of another user for their own benefit.
2. The point-of-interest identifier present in the claim is different from the one present in the visit. *Rationale*: The prover is trying to prove they are at a certain point of interest by submitting evidence that they are at another.
  3. The claim generation timestamp is not within the visit period. *Rationale*: The prover is submitting evidence generated before or after the visit. *Note*: The timestamp present in the claim is collected on the same (prover) device as the timestamps defining the visit period, so they are guaranteed to be synchronized.
  4. The claim is not properly signed by the prover. *Rationale*: The evidence is not recognized as authentic.

If any of these conditions are met we know that the prover is behaving maliciously and as such we automatically **reject** the visit.

**Endorsement validation** Second, we move on to the endorsement validation, where if any of the following conditions are met the endorsement is invalid:

1. There is an error in the decryption or further parsing of the endorsement. *Rationale*: The endorsement was not properly encrypted or its format is invalid.
2. The prover named in the claim and the witness named in the endorsement are the same user. *Rationale*: Either the prover is self-generating endorsements, or the witness maliciously issued the endorsement by naming the prover as witness.
3. The witness named in the endorsement has already issued one of the endorsements accounted for the given visit. *Rationale*: Either the prover is submitting multiple endorsements generated by the same witness deliberately, or the same witness is impersonating several to mislead the prover.
4. The point-of-interest identifier present in the endorsement is different from the one present in the claim. *Rationale*: Either the prover or the witness is lying about their location.
5. The endorsement issuance timestamp is not within the visit period or is less than the claim generation timestamp. *Rationale*: Either the prover is submitting endorsements issued before or after the visit, or the witness is lying about the endorsement issuance timestamp. *Note*: Since the endorsement issuance timestamp is assigned by the witness device and the timestamps defining the visit period and claim generation are assigned by the prover device, we assume that both devices synchronize their clocks periodically and therefore consider a maximum clock skew of 2 minutes.

6. The endorsement is not properly signed by the witness. *Rationale:* The endorsement is not recognized as authentic.

If any of these conditions are met, unlike claim validation, we are not sure who is behaving maliciously, whether the prover or the witness (because the endorsement may have been forged) and as such we simply **ignore** it.

**Challenge response validation** To conclude, the challenge responses are validated, according to Algorithm 1 (the values  $r$ ,  $c$ ,  $a$ ,  $b$  and  $z$  were introduced in Section 4.3.2.2). If there is any response bit  $r_i$  that does not match the expected one, it means that either the user who answered the challenges was not in possession of the secret numbers  $a$  and  $b$ , which indicates that the prover was colluding with another, or the witness tampered with the challenge responses the prover gave, in either case, we **ignore** the endorsement as we fail to recognize its authenticity.

---

**Algorithm 1** Challenge response validation

---

```

for  $i = 0$ ;  $i < numberOfChallenges$ ;  $i ++$  do
  if  $r_i \neq (z_i$  if  $c_i == 1$  else  $a_i)$  then
    Response to challenge  $i$  incorrect  $\rightarrow$  Endorsement is invalid.
  end if
end for

```

---

#### 4.3.3.2 Witness Decay

To protect the system against systematic prover-witness collusion, we implemented a witness decay mechanism whose function is to reduce the weight that endorsements, issued by recurring witnesses of a given prover, have under the strategy confidence calculation.

Thus, the weight of an endorsement is calculated as follows:

$$endorsementWeight(p, w) = \frac{Rw}{Npw + 1} \quad (4.1)$$

where  $Rw$  is the reputation of the witness  $w$  who issued the endorsement and  $Npw$  is the number of routes the prover  $p$  initiated or completed, which the witness  $w$  has already testified to. In this way, the decline in the weight of an endorsement issued by a given witness is inversely multiplicative as a function of the number of routes that contain visits already endorsed by the witness to the prover in question. In other words, a recurring witness increasingly loses weight in the certification of the prover visits.

### 4.3.3.3 Visit Confidence

The confidence assigned to the visit is the product of the confidence that the physical displacement is possible with the sum of the confidence acquired with the different strategies:

$$\min(\text{displacementConfMult} \times (\text{wiFiAPsConf} + \text{peerEndorsementsConf}), 1) \quad (4.2)$$

where the *displacementConfMult* is the confidence multiplier that the displacement between the reported points of interest is physically plausible, the *wiFiAPsConf* is the confidence assigned by the Wi-Fi Scavenging and TOTP strategies (presented in Section 2.1), and the *peerEndorsementsConf* is the confidence assigned by the P2P Witnessing strategy.

**Displacement confidence multiplier** We made an adjustment to the calculation of the confidence assigned to the displacement. First, we calculate the travel speed, which is the division between the distance traveled (calculated by the Haversine formula <sup>6</sup> for the distance between the geocoordinates of the point of interest of the current visit and the last validated visit) and the time the user had to travel that distance (which is the difference between the entry time of the current visit and the exit time of the last validated visit). We define two configurable constants *maxExpectedSpeed* and *maxAdmissibleSpeed* whose respective default value assigned was 100 and 200 km/h. If *travelSpeed*  $\leq$  *maxExpectedSpeed*: 100% displacement confidence is assigned; If *travelSpeed*  $\geq$  *maxAdmissibleSpeed*: 0% displacement confidence is assigned, then the visit is rejected immediately; Otherwise, the displacement confidence is calculated according to the following parabolic function:  $1 - \left(\frac{\text{travelSpeed} - \text{maxExpectedSpeed}}{\text{maxAdmissibleSpeed} - \text{maxExpectedSpeed}}\right)^2$ , shown in the plot in Figure 4.2. This formula was chosen so that the closer the travel speed is to *maxExpectedSpeed*, the slighter the decline, on the contrary, the closer to *maxAdmissibleSpeed*, the sharper the decline. If the travel speed is greater than or equal to 200 km/h, it is physically impossible, hence the value of the multiplier is 0.

---

<sup>6</sup>[https://en.wikipedia.org/wiki/Haversine\\_formula](https://en.wikipedia.org/wiki/Haversine_formula)



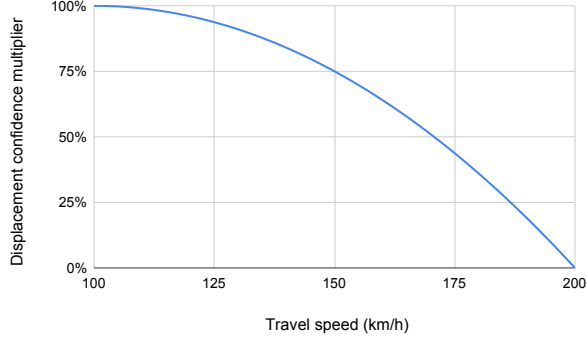


Figure 4.2: Displacement confidence decline according to travel speed.

**Peer endorsements confidence** After all peer endorsements go through the validation process defined above, we request the reputation of the prover and all witnesses who issued valid endorsements to the reputation system [MP22b]. Upon receiving the intervening user reputations (a value in the range of  $[0, 1]$ ), we sum up all weights of the valid endorsements, being the strategy confidence calculated as follows:

$$\min\left(\frac{\sum_w \text{endorsementWeight}(p, w)}{\text{endorsementWeightTarget}}, 1\right) \quad (4.3)$$

such that  $\text{endorsementWeight}(p, w) = \frac{Rw}{Npw+1}$ , and the  $\text{endorsementWeightTarget}$  is the endorsement weights sum value that it is intended to achieve so that the strategy confidence is 100%, value which was defined in Section 5.1.1.1 of the evaluation. The  $Rw$  is managed entirely by the reputation system, the only thing we have to do to update it is to report the users good and bad behavior. For this purpose, when the visit validation: – **succeeds**, it is reported the good behavior of the prover, as well as that of the witnesses who issued valid endorsements (thus helping to certify the visit); – **fails**, it is reported the prover bad behavior that can be considered *intentional* (e.g. claim signature is invalid) or *accidental* (e.g. the acquired visit confidence is less than the confidence threshold), depending on its severity, the prover reputation is decreased more sharply or slightly, respectively. To maintain  $Npw$ , we store the valid endorsements in the database, so we can get it with a simple SQL select query using the **COUNT** aggregate function.

**Confidence threshold adjustment** After calculating the visit confidence, if it is greater than or equal to the confidence threshold, then the visit is accepted, otherwise it is rejected. The confidence threshold (75% default) is adjusted according to the prover reputation. If  $\text{proverReputation} \geq 0,5$  (the prover has a well-behaved track record) the confidence threshold is not adjusted; Otherwise, the confidence threshold is adjusted according to the following

linear function:  $1 - \frac{1 - \text{confidenceThreshold}}{0,5} \times \text{proverReputation}$ , illustrated in the plot in Figure 4.3. In other words, the lower the reputation, the higher the confidence threshold the prover needs to acquire for their visit to be validated.

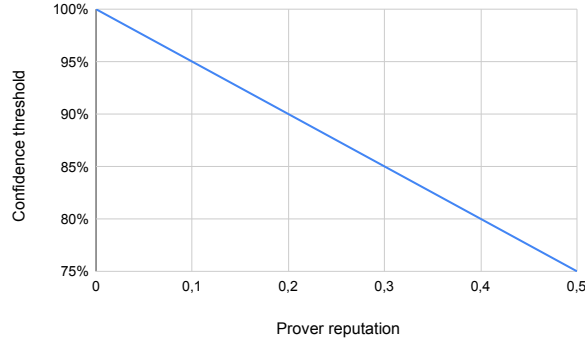


Figure 4.3: Confidence threshold adjustment according to the prover reputation.

#### 4.3.4 Threat Mitigation Assessment

We now explain how each aforementioned threat (presented in Section 4.3.1) is mitigated by this strategy:

- To address **T1**, the sensitive messages, which are claims and endorsements, are not disclosed to eavesdroppers: the claim is not sent to the witnesses, only its signature; the endorsement is sent to the prover encrypted with the server public key; in addition, they are sent to the server through an HTTPS connection that uses the SSL protocol to protect the messages exchanged;
- For **T2**, users cannot take advantage of endorsements issued on behalf of other users as the endorsement contains the signature of the claim which in turn contains the prover identity;
- For **T3**, the endorsements are bound to a location and a timestamp, they can be submitted later, but there is nothing to be gained from keeping them for that purpose, as even if the prover associates the endorsements with a subsequent visit, the server will be able to identify that the endorsement was not issued in the visit period and as such ignores it;
- In the case of **T4**, systematic prover-witness collusion is essentially mitigated by the witness decay mechanism in place, that devalues the weight of endorsements issued by recurring witnesses; but also by requiring a confidence threshold for the visit to be accepted, which is only possible to obtain with a considerable number of witnesses (the better well-behaved track record they have, the less are needed);

- In the case of **T5**, prover-prover collusion is mitigated by the DB protocol, as it allows witnesses to verify if the prover they are communicating with is indeed on site, preventing this prover from channeling messages from another prover that is in another location;
- To thwart **T6**, the presented DB protocol maintains the privacy of users when communicating with each other as all personal sensitive information (such as their username and location) is encrypted with the server public key, moreover, not even the device name is included in the messages exchanged, only the device MAC address, which is temporary.

## 4.4 Gamification

We integrated gamification features into the CROSS City app with these two goals in mind:

1. Increase tourist engagement to not only draw but also retain users in the app;
2. Make tourists voluntarily want to use the P2P Witnessing strategy, since, for its proper functioning, it requires the participation of multiple users.

To this end, we developed the following gamification features designed to provide extrinsic motivation to users:

- **Scoring System and Scoreboard:** To foster competition among app users and encourage the desire to climb to the top;
- **Badge System:** It encourages app users to complete activities and triggers a sense of accomplishment when completing them;
- **In-app Currency:** It increases user in-app loyalty through the potential to save money by completing previous activities.

Furthermore, some adjustments were made in them to reward more generously users who use the P2P Witnessing strategy. Let us now introduce them in depth.

**Scoring System and Scoreboard** The scoring system works as follows: when a user successfully validates a visit, they receive a multiple of 100 experience points (XP), corresponding to the number of previous visits on that same route (e.g. if it is the first visit of the route they will receive 100 XP, if it is the second 200 XP, if it is the third 300 XP, etc.), this XP multiplicity is implemented to encourage users to finish the routes they have initiated, since the points of interest that constitute it, together, are more meaningful. The other way to earn XP is through the use of the P2P Witnessing strategy, more specifically, issuing valid endorsements to other users, whenever the server certifies a visit, 10 XP is awarded to witnesses who issued

valid endorsements, for thus helping the prover to certify their visit. The users are then ranked according to their all-time, seasonal (resetting every 4 months) and weekly (resetting every Monday) score. These scoreboard partitions are to make it more dynamic. The scoreboard screen is displayed in Figure 4.4.

**Badge System** The badge system works as follows: when a user reaches a certain condition for the first time, they receive the corresponding badge. The badge system is designed to be extensible, so that it is simple to add new badges to the database and code the respective achievement conditions for the integration of new badges, but for now we have designed 7 badges, which can be earned: when the user completes their first route; when the user completes their first 3, 15 and 30 visits; and finally, when the user endorses 10, 100 and 1000 users through the P2P Witnessing strategy. To this end, the achievement conditions are checked whenever users submit a successfully validated visit (for the attribution of badges that depend on the number of visits and routes), and whenever the user endorses a successfully validated visit (for the attribution of badges that depend on the number of endorsements). The earned badges screen is displayed in Figure 4.5.

**In-app Currency** The in-app currency is named gems, and they are assigned to users: when they successfully validate a visit; the amount of gems earned varies in multiples of 50, in the same way as the XP earned, taking into account the number of previous visits on that same route (e.g. 50 gems if it is the first visit on the route, 100 gems if it is the second, 150 gems if it is the third, etc.), to motivate users to complete the routes they initiate. The other way to earn gems is by endorsing the visit of other users through the P2P Witnessing strategy, in this way, when the server successfully validates a visit that a certain witness has endorsed, a reward of 5 gems is awarded to that witness. As a means of payment with gems, we developed an in-app card, which displays a QR code with the user JWT, encrypted with the server public key (embedded in it in base 64) so that only the server can retrieve it and a malicious QR code scanner cannot impersonate the user in question. On the other hand, we also developed a scanner app for this card, intended to be used by a point-of-interest clerk to deduct the gems corresponding to the cost of the entrance ticket, the clerk introduces the gems to be deducted, the user presents their in-app card QR code, the scanner app scans it and sends a request to the server containing the encrypted user JWT and the gems to be deducted. Upon receiving the request, the server decrypts the user JWT and decodes it, accessing the identity of the user making the payment (authenticated by the server-issued JWT). In case the user possesses

the gems to be deducted, the server deducts them and notifies the user that the payment was successful, otherwise it responds to the clerk scanner app with an exception stating that the user does not possess the necessary gems to make the payment. The verification of sufficient gems and respective deduction is performed in a database transaction to avoid race conditions and the user getting negative gems by paying multiple tickets at the same time. The in-app card screen is displayed in Figure 4.6.

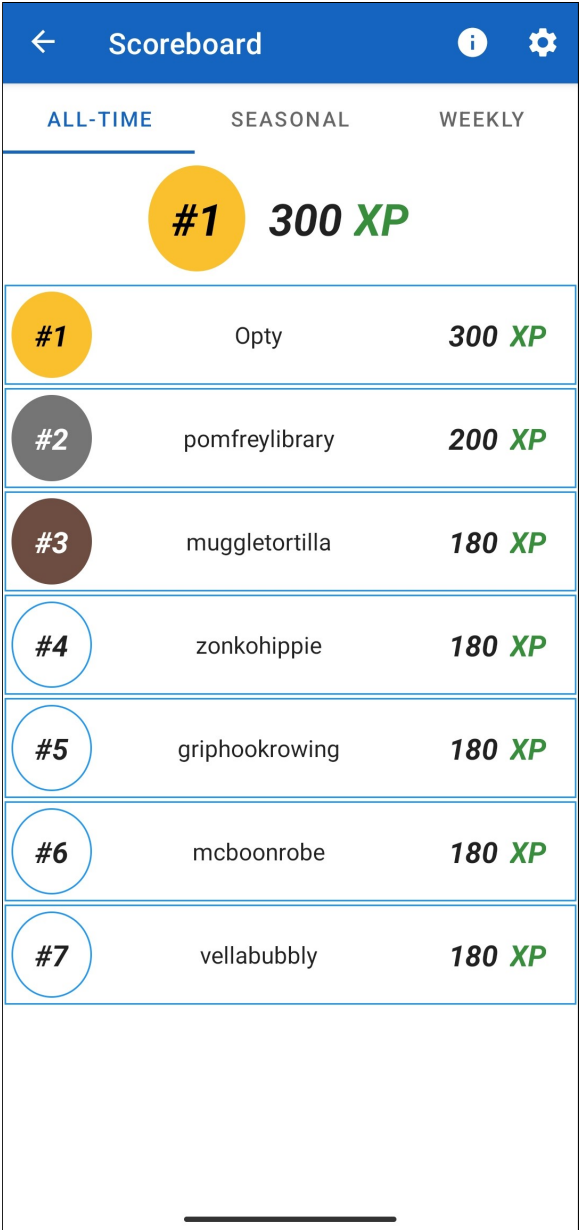


Figure 4.4: All-time scoreboard.

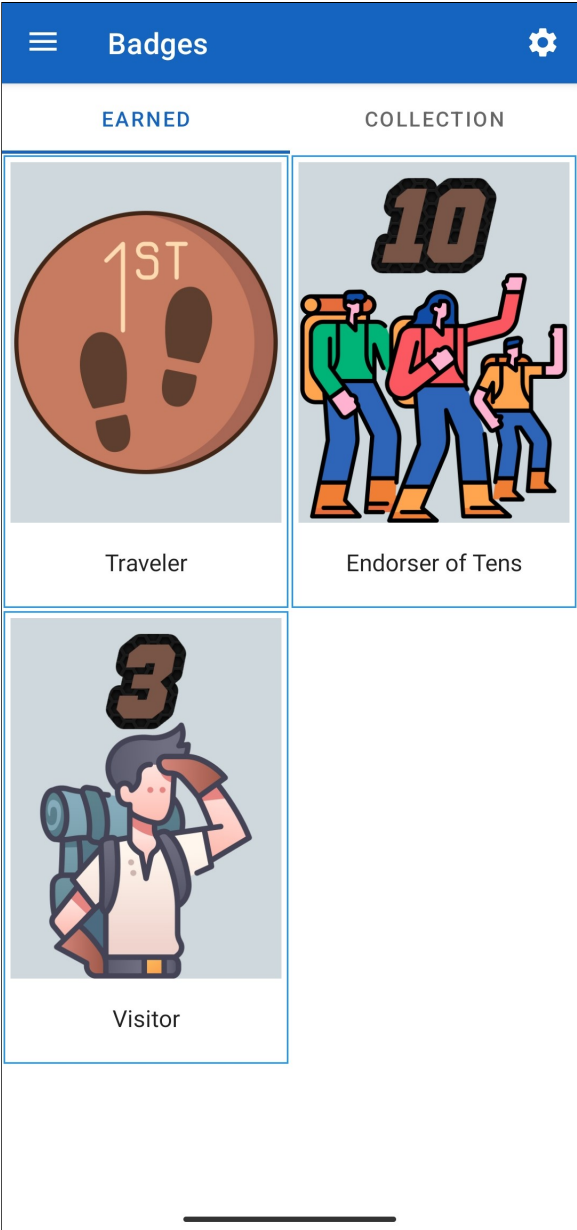


Figure 4.5: Earned badges.



Figure 4.6: In-app card for gem-based payment.

After implementing the different gamification features, we developed the following extensions:

- *User Profile*: We integrated a user profile where the user can see both their ranking on each scoreboard partition and badges earned. Moreover, any user can access the profile of any other user present in the scoreboard by clicking on the corresponding user entry.
- *Chests and Animations*: When a user successfully validates a visit, the corresponding rewards are presented in the form of the contents of an opening chest. All animations were taken from the LottieFiles <sup>7</sup> library (the lottie format was chosen for being lightweight and high-quality) and are rendered natively on the mobile device with the support of the

---

<sup>7</sup><https://lottiefiles.com/featured>

Lottie <sup>8</sup> library.

- *Rewards Notifications*: To provide a more interactive user experience, we made the rewards attribution engine dynamic, since some rewards are assigned without direct communication from the user to the server, such as gems and XP for endorsing peer visits or badges that are awarded after a certain number of endorsements performed. In this case, the server sends a notification to the user via the Firebase Cloud Messaging <sup>9</sup> platform that allows the server to notify a client app at no cost. We also make use of notifications to notify the user of the completion of the gem-based payment for entrance tickets.
- *Help Buttons*: Finally, with the aim of helping users understand the rewards attribution methods, namely the different ways to obtain XP and gems, we developed information and hint buttons.

We also designed more gamification features to provide intrinsic motivation to users as well as user experience features, however, due to time constraints, they are presented as future work in Section 6.2. The ones developed here were prioritized because they are the ones that can be adjusted to reward tourists for using the P2P Witnessing strategy, which is our main goal.

## 4.5 Summary

We implemented the peer-based collusion-resilient location certification strategy (P2P Witnessing). We described its peer endorsement acquisition and validation protocols and explaining how they mitigate the aforementioned threats. This strategy extends the reach of the CROSS City system to locations with no surrounding infrastructure and, furthermore, fortifies its security against malicious users. In addition, we have also implemented gamification elements, namely: scoreboard, badges and in-app currency. These elements aim to provide extrinsic motivation by rewarding users for visiting points of interest but also for using the P2P Witnessing strategy, as they are intended to engage and encourage users to use the app and adopt this strategy, as it relies greatly on user participation.

---

<sup>8</sup><https://github.com/airbnb/lottie-android>

<sup>9</sup><https://firebase.google.com/docs/cloud-messaging/android/client>





# Chapter 5

## Evaluation

In this Chapter we present the evaluation of this work, which is divided into two parts: first, we evaluate the *P2P Witnessing* strategy in laboratory experiments to assess its effectiveness, responsiveness and collusion-resilience; second, we evaluate its feasibility in a real-world scenario and the impact of gamification on user motivation.

### 5.1 P2P Witnessing Strategy Experiments

The P2P Witnessing strategy relies on nearby peers, acting as witnesses, to endorse user visits to points of interest. This strategy must therefore, in addition to being effective and responsive, be resilient against malicious users who engage in prover-witness and prover-prover collusion. These requirements are easier to assess in controlled environments, where we know exactly what is happening and under what conditions. We will first assess the components related to the validation of peer endorsements (on the server), and then their acquisition (on the mobile application).

#### 5.1.1 Peer Endorsement Validation Assessment

In this Section we evaluate the validation of peer endorsements. This system component is implemented on the server, which enables us to use a Java testing client instead of the mobile application itself to submit visits, so that we have more flexibility to simulate a palette of users and scenarios. In a first phase of this evaluation, we estimate the value of the parameters necessary for the validation of peer endorsements; then we assert the resilience of the strategy against prover-witness collusion; finally, we assert the responsiveness of the validation protocol.

### 5.1.1.1 Parameter Values Estimation

For a visit to be successfully validated, it needs to reach a confidence level (percentage) greater than a threshold that is defined by the system operator for the specific route. The threshold varies according to the prover reputation (the lower the reputation, the higher the threshold). The visit confidence calculation is presented in Section 4.3.3.3, but we recap the Equation 4.3 for the strategy confidence calculation that contains the *endorsementWeightTarget* parameter yet to be defined, which is the sum of the weights that is necessary to achieve so that the strategy confidence is 100%, its estimation/selection was made based on the Table 5.1 we compiled.

Table 5.1: Sum of the weights of endorsements that witnesses issue, with variant cardinality and reputation (through previously successfully validated visits); the colors are used just to differentiate the minimum reputation for the prover to successfully validate the visit

<i>Endorsement weight target:</i> 2,1		# Witness endorsements									
		1	2	3	4	5	6	7	8	9	10
# Visits	0	0,35	0,70	1,05	1,40	1,75	2,10	2,45	2,80	3,15	3,50
	1	0,48	0,96	1,44	1,92	2,40	2,88	3,36	3,84	4,32	4,80
	2	0,55	1,10	1,65	2,20	2,75	3,30	3,85	4,40	4,95	5,50
	3	0,60	1,20	1,80	2,40	3,00	3,60	4,20	4,80	5,40	6,00
	4	0,64	1,28	1,92	2,56	3,20	3,84	4,48	5,12	5,76	6,40
	5	0,67	1,34	2,01	2,68	3,35	4,02	4,69	5,36	6,03	6,70
	6	0,69	1,38	2,07	2,76	3,45	4,14	4,83	5,52	6,21	6,90
	7	0,72	1,44	2,16	2,88	3,60	4,32	5,04	5,76	6,48	7,20
	8	0,74	1,48	2,22	2,96	3,70	4,44	5,18	5,92	6,66	7,40
	9	0,75	1,50	2,25	3,00	3,75	4,50	5,25	6,00	6,75	7,50
		Prover Reputation = 0		=>	Confidence Threshold = 100%						
		Prover Reputation ≥ 0,35		=>	Confidence Threshold ≥ 82,5%						
		Prover Reputation ≥ 0,5		=>	Confidence Threshold ≥ 75%						

In Table 5.1 we can see in red, yellow and green the sum of endorsements weights that a visit would have to reach to be successfully validated if the prover had respectively a reputation: close to 0 (worst possible behavior); greater than or equal to 0,35 (reputation of a newly created user); greater than or equal to 0,5 (reputation of a well-behaved user). This is assuming that the minimum confidence threshold set by the system operator is 75%. All witnesses, who endorse the visit, in each row of the table have the same reputation equivalent to the number of visits (to points of interest) previously validated, assuming they never behave misbehave. For each visit carried out by the witnesses, we assumed that there would be 6 reports of good behavior to the reputation system: 1 corresponding to the successful validation of the visit; 5 if the witness comes across 5 other users during the visit and endorses their claim.

The *endorsementWeightTarget* value of 2,1 was chosen considering the following scenario, making an adhoc assumption that there are between 3 and 6 witnesses at the points of interest at the time of visit: if the prover has been misbehaving, it is necessary to find as many witnesses as possible, or a smaller number of witnesses but with a well-behaved track record; if the prover has been behaving well, we want this same value to be lower, in order to comply with the **G1** guideline. This way, if the prover has been misbehaving, 6 witnesses are needed, or 3 honest witnesses with 7 pre-validated visits; if the prover has been behaving well, either only 5 witnesses are needed, or the same 3 honest witnesses, but with only 2 pre-validated visits, thus fulfilling the **G1** guideline. This low number of witnesses does not pose a threat of systematic collusion, because we have a witness decay mechanism (presented in Section 4.3.3.2) that makes it unfeasible to systematically get a witness who has not yet been affected by this decay with such a well-behaved track record. This would be forcing the attackers to make themselves that number of pre-validated visits with that number of user accounts whenever they wanted to carry out this attack.

#### 5.1.1.2 Prover-Witness Collusion-Resilience

Prover-witness collusion could be carried out by a prover that has available a set of witnesses who can endorse the visit, even if they are not physically at the point of interest. To assess the resilience of the system against systematic prover-witness collusion, we need to answer the following question: *How many successful collusions can a prover perform with  $N$  newly created witnesses?* To find an answer, we generated a series of plots like the one in Figure 5.1, one for each number of witnesses (1 to 10) in the set of newly created witnesses at the disposal of the prover, where we compare the acquired confidence with the confidence threshold necessary to obtain for the visit to be accepted, attempting to carry out up to 10 collusions with the same set of witnesses endorsing the visits. In these plots, the confidence acquired in the first collusion attempt can be derived from Equation 4.1 as  $\min(\sum_w Rw, 1)$ , so the greater the number of witnesses, the greater the initial acquired confidence. On subsequent attempts it drops sharply until the claim is no longer accepted, while the confidence threshold increases linearly with the number of rejected claims. This drop in the confidence acquired results from the witness decay mechanism in place (presented in Section 4.3.3.2), which multiplicatively reduces the weight of witness endorsements the more the witness issues on behalf of the prover in question.

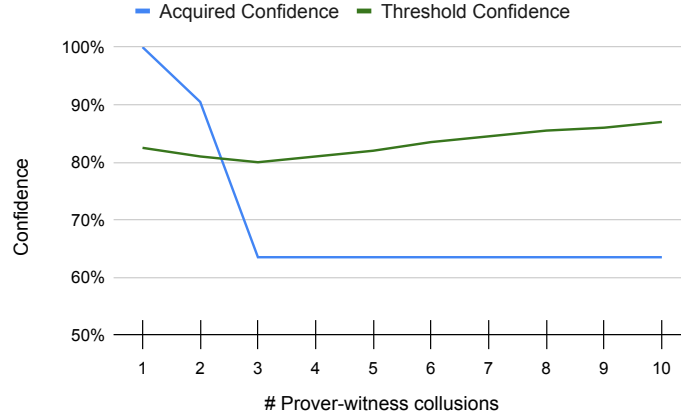


Figure 5.1: This plot was generated attempting to perform multiple collusions with 10 newly created witnesses, we can see that even with 10, only 2 collusions are possible to perform.

From the analysis of these plots, it was possible to arrive at Table 5.2 which summarizes the number of prover-witness collusion accepted by the system given the number of newly created witnesses the prover controls.

Table 5.2: Number of collusions possible to perform with the specified number of newly created witnesses.

# Witnesses	#Collusions
[1, 4]	0
[5, 8]	1
[9, 10]	2

We can see that, if the malicious prover has up to 4 newly created witnesses at their disposal, they still will not be able to forge their visit through collusion. In case they control a greater number of witnesses, then they will be able to carry out the attack, but only a reduced number of times. Thus, we conclude that the system meets the **G2** guideline, being resilient against systematic prover-witness collusion. Even if witnesses have a well-behaved track record, since the decay applied to the reputation of the witness is independent of their behavior as defined in Equation 4.1.

### 5.1.1.3 Validation Protocol Responsiveness

We now evaluate the responsiveness of the peer endorsement validation protocol to assess whether the server is able to provide real-time feedback to the user when submitting a visit.

**Setup** The requests in these experiments was made from a computer other than the one on which the server was running, through a private wired network, so that the testing simulation would not computationally overload the server. The computer on which the server was running was a GL75 Leopard 10SEK model equipped with an Intel<sup>®</sup> Core<sup>™</sup> i7-10750H CPU Hexa-Core 2.60GHz, with 16GiB of RAM. The computer running the simulation is a GL553VW model equipped with an Intel<sup>®</sup> Core<sup>™</sup> i7-6700HQ CPU Quad-Core 2.60GHz, with 16GiB of RAM. Both running on top of Ubuntu 20.04.4 OS. The times were recorded in the testing client.

To assess the scalability and responsiveness of the peer endorsement validation, we answer the following questions: 1. *How long does it take the server to validate a visit endorsed by  $N$  witnesses?* 2. *What is the respective overhead and likelihood of the occurrence of  $N$  simultaneous visit submissions?*

To answer question 1, we performed the experiments and the results are shown in Figure 5.2 where we can see the evolution of the visit validation time in the server, varying the number of witnesses (1 to 30, considering a scenario where the user crosses paths with 1 witness every 30 seconds in a 15-minute visit) who endorse the visit.

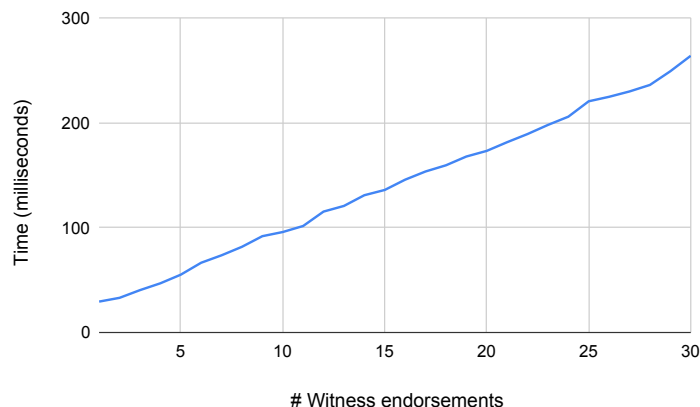


Figure 5.2: Visit validation time varying the number of witnesses who endorse the visit.

From this plot, we conclude that the visit validation time *varies linearly* with the number of peer endorsements present in it, which is to be expected, considering that the validation of an endorsement implies its decryption through an asymmetric cipher and validation of its signature, which is computationally the heaviest component of the visit validation, along with the reputation system overhead that is also more noticeable the greater the number of witnesses. We can deduce from the plot a validation time function  $t(w) \approx 8,09w + 29,43$ , where  $t(w)$  is the expected validation milliseconds of a visit endorsed by  $w$  witnesses. Considering that the protocol provides a responsive feedback experience to the user, if the validation response time is

less than 1 second, according to  $t(w)$  this would only happen for  $w \geq 120$ , being quite unlikely that this number of witnesses would be obtained we conclude that in this scenario the protocol is responsive.

To answer question 2, we performed the experiments and the results are shown in Figure 5.3 where we vary the number of simultaneous submissions of visits endorsed by 15 witnesses (the average number considered in the previous experiment). The maximum number of simultaneous visits considered is 5, because in a scenario where our app is used by 10 thousand users simultaneously making a visit, assuming an average visit duration of 15 minutes, the likelihood of more than 5 users submitting the visit within the same validation interval (135,92 milliseconds, which is the validation time recorded in the previous experiment for a visit endorsed by 15 witnesses) is  $\approx 0,46\%$  following a binomial distribution <sup>1</sup> with success probability  $p = \frac{135,92 \text{ millis}}{15 \text{ minutes}} \approx 0,02\%$  with  $n = 10$  thousand attempts. The binomial distribution fits our use case as it is considered an equal success probability in all our experiments as they can be considered independent.

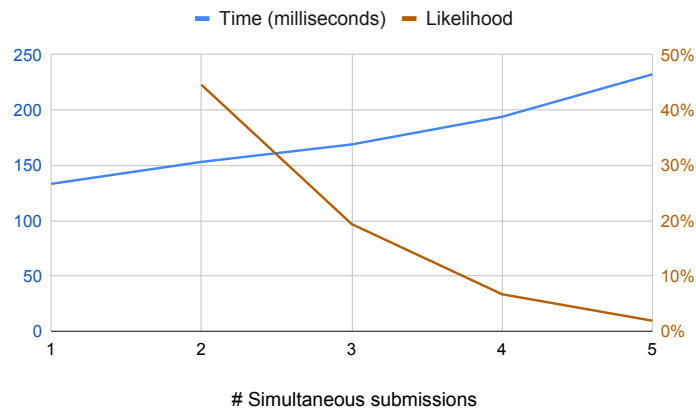


Figure 5.3: Visit validation time varying the number of simultaneous visit submissions.

In Figure 5.3, we can see that, as expected, the visit validation time increases with the number of simultaneous submissions, however, given an increase ratio of only  $\frac{232,28}{133,25} = 1,74$  (comparing the validation response times of 1 and 5 simultaneous submissions) and the likelihood of that number of simultaneous submissions occurring, we conclude that submission concurrency is not a factor of concern for the protocol responsiveness. With this, we conclude that the system is responsive meeting the **G3** guideline, even in cases where multiple visits are submitted simultaneously.

<sup>1</sup><https://fenix.tecnico.ulisboa.pt/downloadFile/1970943312285264/2017-02-TabelasEstatisticas-PE.pdf>

## 5.1.2 Peer Endorsement Acquisition Assessment

This Section assesses the acquisition of peer endorsements, and it comprises of: the resilience of the system against prover-prover collusion; the feasibility of the protocol given its performance; and finally, the impact of adding this protocol on the power consumption of the mobile device.

**Setup** For these tests, two different mobile devices were used, which communicate with each other over BLE. The results of both devices were collected to observe their discrepancies. These mobile devices were: a Samsung Galaxy S9 <sup>2</sup> model equipped with an Exynos 9810 CPU Quad-Core 2.8GHz + Quad-Core 1.7GHz, 4GiB of RAM, on top of Android 10 OS; and a Xiaomi POCO X3 Pro <sup>3</sup> model equipped with a Qualcomm<sup>®</sup> Snapdragon<sup>™</sup> 860 CPU Octa-Core up to 2.96GHz, 8GiB of RAM, on top of Android 11 OS.

### 5.1.2.1 Prover-Prover Collusion-Resilience

Prover-prover collusion could be carried out by two provers that are far from each other and one of them forwards the DB protocol challenges to the one that is elsewhere, managing to gather endorsements that certify that the other prover is on site, when in fact it is not. We are preventing this by restricting the challenge response times that are acceptable for a witness to endorse a claim to be less than the time required for the *witness*  $\leftrightarrow$  *prover* <sub>$\alpha$</sub>  communication over BLE plus the *prover* <sub>$\alpha$</sub>   $\leftrightarrow$  *prover* <sub>$\beta$</sub>  communication over Wi-Fi.

To assess the resilience of the system against prover-prover collusion, we answer the following question: *What is the DB protocol acceptance rate with honest provers versus colluding provers?* To look for an answer, we start by monitoring the average challenge response times (taking a representative number of 50 samples from each mobile device to account for the volatility of the execution environment) between the two mobile devices in three scenarios: when separated by 1 meter, 10 meters (to see how times change depending on the distance between the two devices), and in adjacent rooms with the two doors closed (to see how times change with obstacles present between the two devices). The two mobile devices were periodically (approximately every 15 seconds) exchanging endorsements with each other, and at each average challenge response time recorded, 64 challenges are exchanged, discarding the 10% worst, to reduce underlying noise. The times recorded in each scenario were inserted in the plots in Figures 5.4 to 5.6.

---

<sup>2</sup>Samsung Galaxy S9: abbreviated as S9 in the following graphs and tables.

<sup>3</sup>Xiaomi POCO X3 Pro: abbreviated as X3 in the following graphs and tables.

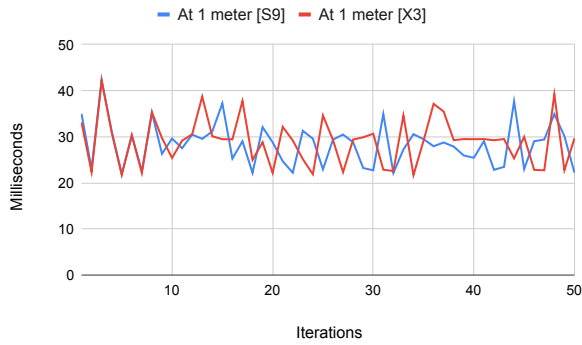


Figure 5.4: Average challenge response times with the devices 1 meter apart.

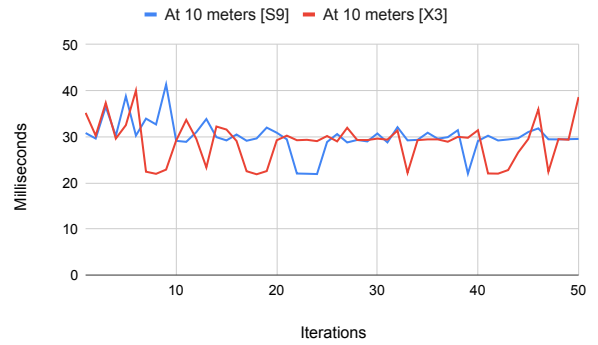


Figure 5.5: Average challenge response times with the devices 10 meters apart.

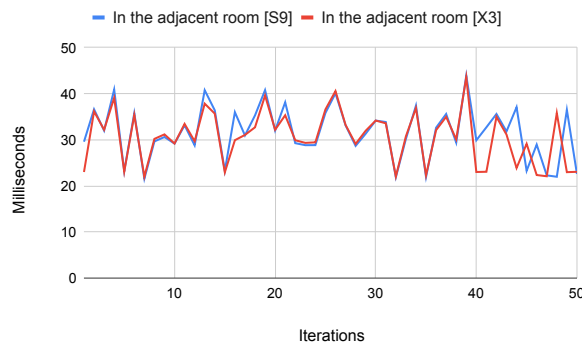


Figure 5.6: Average challenge response times with the devices in adjacent rooms.

We can see in these plots that regardless of the arrangement of the two mobile devices, the times oscillate between approximately the same values, i.e. the message propagation time does not play a major role in the challenge response time fluctuation, nor does the response bit calculation as we have seen that it takes no longer than  $\approx 1$  millisecond, excluding parts, the component that plays the most significant role in this context must be the delay until the message is transmitted, this is because the DB protocol is developed entirely at the application layer as we want the app to work on any unmodified smartphone.

To get a better view of the distribution of the average challenge response times recorded, we gathered all the times obtained across all scenarios (since they oscillate similarly between the same values) and inserted them in the plot in Figure 5.7.



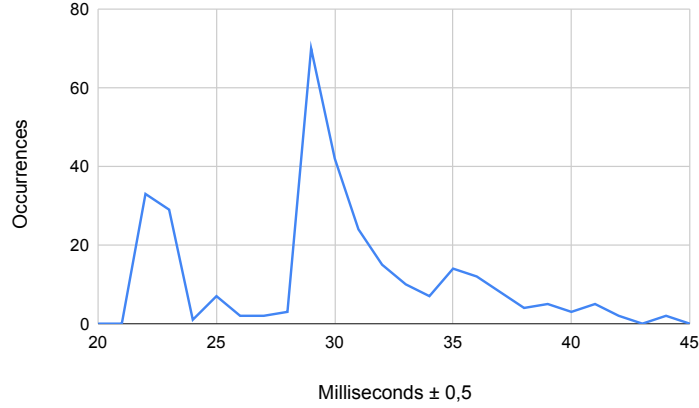


Figure 5.7: Frequency of average challenge response times.

As we can see, this graph does not seem to follow any distribution model, so it is not possible to derive a probability function associated with an existing model. Alternatively, we consider that the probability of a user responding to challenges in time  $T$  is given by the function  $P(t \leq T) = \frac{\sum_t (1 \text{ if } t \leq T \text{ else } 0)}{N}$  such that  $t$  are the times collected from the sample collected above and  $N = 300$  the number of samples. This probability function will be used next to calculate the *acceptance* and *rejection rate* as a function of challenge response times.

Now that we have measured the time distribution of the *witness*  $\leftrightarrow$  *prover* $_{\alpha}$  communication over BLE, let us estimate the *prover* $_{\alpha}$   $\leftrightarrow$  *prover* $_{\beta}$  communication time over Wi-Fi. In other words, let us measure the latency introduced in challenge responses by a user attempting to perform prover-prover collusion. To do so, we recorded the average ping time from Cascais (Portugal) to some locations <sup>4</sup> in Portugal and Spain, Portugal’s neighboring country, (calculated similarly to the average challenge response times, i.e. recording 64 ping times and discarding the worst 10%) for each iteration (10), made at different times of the day, on both mobile devices in parallel. From this times (included in Table 5.3), we conclude that they vary depending on the mobile device, but not on the time of the day, as their standard deviation recorded in each iteration is small, so this reconfiguration would only be necessary to repeat if either the mobile devices network capabilities or the network speed are upgraded.

<sup>4</sup>The IPs used to ping each location were: 193.136.128.169 (Lisbon, Portugal); 193.137.35.140 (Porto, Portugal); 5.134.119.53 (Madrid, Spain); 185.166.215.231 (Barcelona, Spain).

Table 5.3: Average ping time per challenge statistics; Pings from Cascais, Portugal to the mentioned locations.

Average ping times statistics (milliseconds)	Lisbon, Portugal				Porto, Portugal				Madrid, Spain				Barcelona, Spain			
	Wi-Fi		4G		Wi-Fi		4G		Wi-Fi		4G		Wi-Fi		4G	
	S9	X3	S9	X3	S9	X3	S9	X3	S9	X3	S9	X3	S9	X3	S9	X3
Average	8,02	13,68	19,09	21,04	12,84	18,08	24,12	24,79	23,85	29,71	29,73	29,62	26,93	32,69	34,50	36,54
Standard deviation	0,23	1,14	0,47	0,32	0,44	1,08	0,87	0,37	0,82	2,59	1,68	0,87	0,70	0,62	0,33	0,26
Total average	10,85		20,07		15,46		24,45		26,78		29,67		29,81		35,52	

Now that we know about the distribution of challenge response times and the average time a packet takes to reach different locations on different mobile devices, we can now derive the DB protocol acceptance rate in the scenarios incorporated in Table 5.4.

Table 5.4: Witness acceptance rate of honest and colluding provers claims, given the threshold 34.5 milliseconds (explained below how it was defined).

Acceptance rate Threshold: 34,5 millis	Honest	Prover-prover collusion											
		Lisbon, Portugal						Porto, Portugal					
		Wi-Fi			4G			Wi-Fi			4G		
		S9	X3	Average	S9	X3	Average	S9	X3	Average	S9	X3	Average
Attempts: 1	81,67%	24,00%	0,00%	<b>12,00%</b>	0,00%	0,00%	<b>0,00%</b>	0,33%	0,00%	<b>0,17%</b>	0,00%	0,00%	<b>0,00%</b>
Attempts: 2	96,64%	42,24%	0,00%	<b>22,56%</b>	0,00%	0,00%	<b>0,00%</b>	0,67%	0,00%	<b>0,33%</b>	0,00%	0,00%	<b>0,00%</b>
Attempts: 3	99,38%	56,10%	0,00%	<b>31,85%</b>	0,00%	0,00%	<b>0,00%</b>	1,00%	0,00%	<b>0,50%</b>	0,00%	0,00%	<b>0,00%</b>

Remember that if something goes wrong and the endorsement gets rejected users have up to 3 attempts (empirically defined) to request an endorsement from each of the witnesses around them. To calculate the first attempt acceptance rate, this is equal to  $AR_1 = P(t \leq T_{max} - L)$  ( $P$  being the probability function presented above) given that  $AR_i$  is the acceptance rate of attempt  $i$ ,  $T_{max}$  is the threshold value of the average challenge response times, and  $L$  is the communication latency between the two colluding provers ( $L = 0$  if the prover is honest). We set  $T_{max} = 34,5$  milliseconds, which seemed the most appropriate, comparing the acceptance rate of honest and colluding provers, and  $L$  is the ping time from Cascais (Portugal) to the mentioned locations presented in Table 5.3. The acceptance rate of subsequent attempts follows a binomial distribution <sup>1</sup> with a success probability  $p = AR_1$ , the binomial distribution fits our use case as it is considered an equal success probability in all our experiments as they are independent.

As we can see with just 2 attempts, an honest prover is practically guaranteed to get the endorsement successfully (acceptance rate  $\approx 97\%$ ), while in a scenario of prover-prover collusion where one is in Cascais (Portugal) and the other in Lisbon (Portugal), if they are connected over Wi-Fi, the colluding prover on average will only be able to get endorsements from  $\approx 1$  out

of 3 witnesses (acceptance rate for this average case with 3 attempts is  $\approx 32\%$ ); if connected over 4G the colluding prover will not be able to get any. In the scenario in which the colluding provers are further away, such as one in Cascais (Portugal) and the other in Porto (Portugal), even connected over Wi-Fi, it is practically impossible (max acceptance rate with 3 attempts is 1%) for the colluding prover to be able to deceive any witness into issuing an endorsement.

With this, we conclude that the system meets the **G4** guideline, being resilient against prover-prover collusion attacks in the case where the prover is connected over 4G; or over Wi-Fi where they are far from each other (as from Cascais to Porto). Even in cases where the prover is connected over Wi-Fi and both are close to each other, the DB protocol substantially decreases the effectiveness of this attack depending on the mobile device and the latency of the Wi-Fi network.

### 5.1.2.2 Acquisition Protocol Performance

To assess the performance and feasibility of the peer endorsement acquisition protocol, we answer the following question: *How long does the peer endorsement acquisition protocol take to be executed, accounting for connection time, connection setup time, and endorsement time (from the claim is sent until the endorsement is received)?* To investigate it, we monitored the execution time of the peer endorsement acquisition protocol on two mobile phones that periodically (every 15 seconds) issued endorsements to each other (also taking 50 samples from each mobile device).

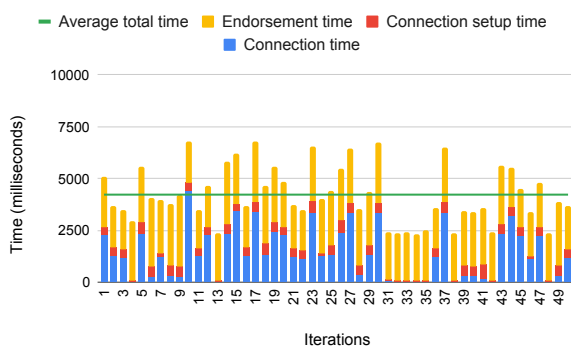


Figure 5.8: Execution time of the endorsement acquisition protocol on Samsung Galaxy S9.

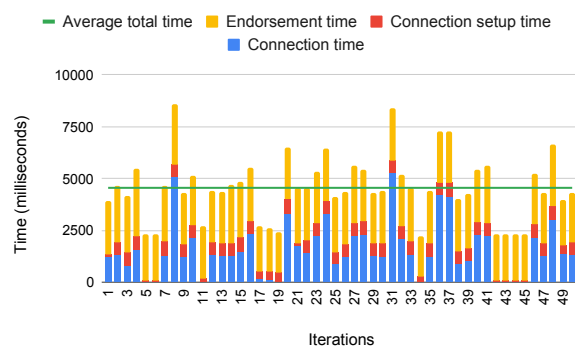


Figure 5.9: Execution time of the endorsement acquisition protocol on Xiaomi POCO X3 Pro.

The samples collected on Samsung Galaxy S9 are shown in the plot in Figure 5.8, those collected on Xiaomi POCO X3 Pro are shown in Figure 5.9. In them we can see that the

average execution time of the peer endorsement acquisition protocol is around 4,5 seconds on both mobile devices, and that the endorsement time is relatively constant, in contrast to the connection establishment and setup time where volatility is most noticeable (reaching 0 in few experiments, in others taking half the total time). To better understand the distribution of time in the different phases of the protocol execution, we can look at Table 5.5 where we can see that the values and respective weights in the protocol execution time recorded in the two mobile devices are similar to each other.

Table 5.5: Distribution of the different components of the protocol execution time (connection establishment time, connection setup time, and endorsement time).

Protocol runtime	Connection time			Connection setup time			Endorsement time		
	S9	X3	Average	S9	X3	Average	S9	X3	Average
Time (milliseconds)	1 432,25	1 564,63	<b>1 498,44</b>	338,12	485,15	<b>411,64</b>	2 451,83	2 500,73	<b>2 476,28</b>
Distribution	33,92%	34,38%	<b>34,16%</b>	8,01%	10,66%	<b>9,38%</b>	58,07%	54,95%	<b>56,45%</b>

Overall, the times obtained appear to be extensible to similar mobile devices and indicate that the peer endorsement acquisition protocol is responsive and does not require the prover and witness to stay close to each other for a long time, fulfilling the **G5** guideline.

### 5.1.2.3 Protocol Impact on the Mobile Device Power Resources

To assess the impact that peer endorsement acquisition has on mobile device power resources, we answer the following question: *How much battery does the acquisition protocol execution consume?* To look into the issue, we profiled the battery usage of the two mobile devices with Batterystats and Battery Historian <sup>5</sup>, simulating a 15-minute visit to a point of interest, comparing the data obtained in 2 scenarios: in the first, only using the Wi-Fi-based strategies, where the access points located nearby are scanned every 30 seconds; in the second, adding to the top the P2P Witnessing strategy, where every 30 seconds we simulate the user crossing with another, making them exchange endorsements with each other.

We observed that the battery consumed by the app in both scenarios on both mobile devices is around the same percentage  $\approx 0.15\%$ , which allows us to conclude that the computation necessary for the exchange of peer endorsements, being performed every 30 seconds, does not represent a noticeable factor in battery consumption. However, in the second scenario, an additional percentage of battery is consumed by Bluetooth  $\approx 0.05\%$ , which is still quite low, as it is designed to be as we are using its LE (Low-Energy) variant. We conclude, therefore, that

<sup>5</sup><https://developer.android.com/topic/performance/power/setup-battery-historian>

the overhead induced by the addition of the P2P Witnessing strategy is not a factor of concern for the user experience, thus fulfilling the **G6** guideline.

## 5.2 Evaluation with Users

In this Section we present the results of the route that we set up through the Alameda campus of Instituto Superior Técnico, comprising: the analysis of the metrics collected, through the instrumentalization of the server and the app, during the visits, to assess the feasibility of the *P2P Witnessing* strategy in a real-world setting; as well as the results of the questionnaire (presented in Appendix C) that aims to assess the gamification impact on the app from the user perspective.

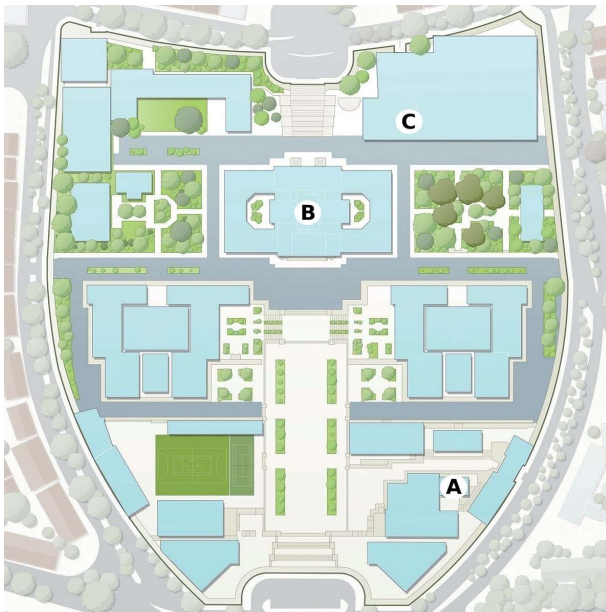


Figure 5.10: Location map of points of interest that compose the tour route.

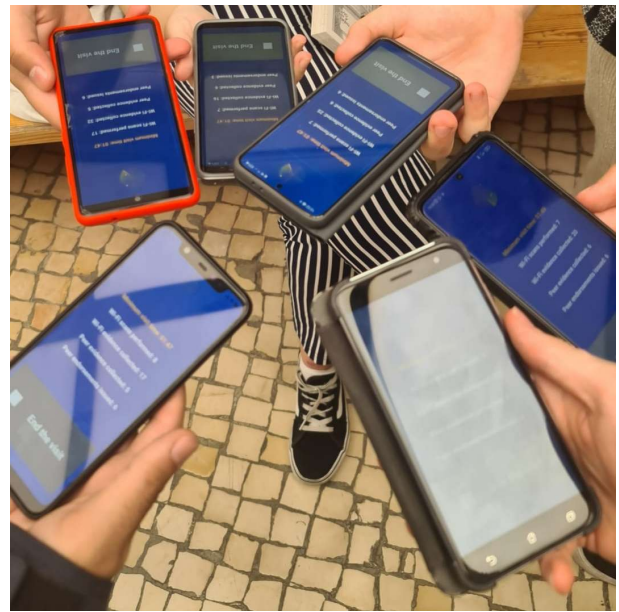


Figure 5.11: Group photo on the tour.

**Route setup** The route was made up of three pavilions, about 200 meters apart: *Informatics Pavilion III* (point A on the map in Figure 5.10), *Central Pavilion* (point B), and *Civil Pavilion* (point C); with each visit lasting 5 minutes. We went to these visit points the day before to scan the access points there and include them in the server database. On the day of the tour, August 2nd of 2022, we deployed the server [MP22a] and the reputation system [MP22b] on Google Cloud, generated the app APK and made it available to a group of 7 users who were students from Instituto Superior Técnico. The 7 users were able to install the app APK, from a

link provided by us, without any additional steps, on their unmodified Android device, fulfilling **G9**. Note that we instrumented the server and app code to print relevant metrics into files, which were collected at the end of the experiments.

### 5.2.1 Instrumentalization Metrics Analysis

Before the main experiment with 7 users, we tested the system with 3 users to assess how the system behaves with a smaller number of witnesses in a real-world setting. At the beginning of each route, users created a new account, as such, they have no behavior history and each user had the same initial reputation. We also asked users to let both location certification strategies enabled (thus leveraging the surrounding Wi-Fi infrastructure as well as the on-site peers).

Table 5.6: Metrics of the route through the Alameda campus with 3 users; the users are arranged in the ascending temporal order of their visit submission; the colors assigned to each user serve solely to better distinguish the respective rows.

<b>Visit 1: IT Pavilion III</b>							
<i>Traveler</i>	<i>Non-Endorsed User Identifiers</i>	<i>N° Endorsements Collected</i>	<i>Endorsement Weights Sum</i>	<i>P2P Witnessing Strategy Confidence</i>	<i>Wi-Fi Scavenging Strategy Confidence</i>	<i>Combined Confidence Acquired</i>	<i>Confidence Threshold</i>
<i>User_1</i>	{}	2	0,70	33,3%	77,3%	100,0%	82,5%
<i>User_2</i>	{}	2	0,76	36,2%	81,8%	100,0%	81,0%
<i>User_3</i>	{}	2	0,80	38,1%	86,4%	100,0%	80,0%
<b>Visit 2: Central Pavilion</b>							
<i>Traveler</i>	<i>Non-Endorsed User Identifiers</i>	<i>N° Endorsements Collected</i>	<i>Endorsement Weights Sum</i>	<i>P2P Witnessing Strategy Confidence</i>	<i>Wi-Fi Scavenging Strategy Confidence</i>	<i>Combined Confidence Acquired</i>	<i>Confidence Threshold</i>
<i>User_2</i>	{}	2	0,42	20,0%	80,4%	100,0%	79,0%
<i>User_3</i>	{}	2	0,44	21,0%	64,7%	85,7%	78,0%
<i>User_1</i>	{}	2	0,46	21,9%	76,5%	98,4%	77,0%
<b>Visit 3: Civil Pavilion</b>							
<i>Traveler</i>	<i>Non-Endorsed User Identifiers</i>	<i>N° Endorsements Collected</i>	<i>Endorsement Weights Sum</i>	<i>P2P Witnessing Strategy Confidence</i>	<i>Wi-Fi Scavenging Strategy Confidence</i>	<i>Combined Confidence Acquired</i>	<i>Confidence Threshold</i>
<i>User_1</i>	{}	2	0,48	22,9%	73,3%	96,2%	76,0%
<i>User_2</i>	{}	2	0,49	23,3%	70,0%	93,3%	75,5%
<i>User_3</i>	{}	2	0,50	23,8%	56,7%	80,5%	75,0%

The metrics collected during the route with 3 users are included in Table 5.6. As we can see, all users endorsed everyone on every visit, so the peer endorsement acquisition protocol, which includes the DB protocol, performed optimally. The users are arranged in the table in the ascending temporal order of their visit submission, therefore endorsement weights are increasing (with the exception of the decline from visit 1 to visit 2) because the user who submits it later has as witnesses the users who submitted it previously, which as such have a superior reputation (recall that the user reputation increases when they successfully validate their visit or when the server successfully validates a visit endorsed by the user in question). The decline in the en-

endorsement weights sum from visit 1 to visit 2 is explained by the witness decay mechanism, since according to Equation 4.1, the witnesses reputation will be divided by 2, since  $Npw$  becomes equal to 1 with the first route endorsement issued by all users to all, on visit 1. Recall that the P2P Witnessing strategy confidence is calculated, according to Equation 4.3, by dividing the endorsement weights sum by the  $endorsementWeightTarget = 2, 1$ . The combined confidence acquired is calculated by adding (+) the confidence acquired by the P2P Witnessing and Wi-Fi-based strategies, according to Equation 4.2, since the displacement confidence multiplier was always equal to 1. The confidence threshold for the visit to be accepted is consistently decreasing as the traveler reputation increases (confidence threshold adjustment explained in Section 4.3.3.3), until it stagnates at 75% (set threshold). As the confidence acquired is greater than the confidence threshold on all visit submissions, all of them were successfully validated. In conclusion, given these metrics, the first experiment with 3 users was a success.

Table 5.7: Metrics of the route through the Alameda campus with 7 users.

<b>Visit 1: IT Pavilion III</b>							
Traveler	Non-Endorsed User Identifiers	N° Endorsements Collected	Endorsement Weights Sum	P2P Witnessing Strategy Confidence	Wi-Fi Scavenging Strategy Confidence	Combined Confidence Acquired	Confidence Threshold
User_1	{}	6	2,10	100,0%	95,5%	100,0%	82,5%
User_2	{}	5	1,90	90,5%	100,0%	100,0%	81,0%
User_3	{}	5	2,00	95,2%	81,8%	100,0%	80,0%
User_4	{}	4	1,68	80,0%	86,4%	100,0%	79,0%
User_5	{2, 3, 4}	6	2,62	100,0%	72,7%	100,0%	81,0%
User_6	{4}	6	2,70	100,0%	100,0%	100,0%	78,0%
User_7	{}	6	2,80	100,0%	90,9%	100,0%	76,0%
<b>Visit 2: Civil Pavilion</b>							
Traveler	Non-Endorsed User Identifiers	N° Endorsements Collected	Endorsement Weights Sum	P2P Witnessing Strategy Confidence	Wi-Fi Scavenging Strategy Confidence	Combined Confidence Acquired	Confidence Threshold
User_1	{}	5	1,22	58,1%	76,7%	100,0%	75,5%
User_4	{}	5	1,49	71,0%	63,3%	100,0%	75,0%
User_5	{1, 4}	6	1,55	73,8%	50,0%	100,0%	78,0%
User_7	{}	6	1,55	73,8%	56,7%	100,0%	75,0%
User_6	{}	6	1,59	75,7%	80,0%	100,0%	75,0%
User_2	{}	6	1,86	88,6%	60,0%	100,0%	75,0%
User_3	{}	6	1,90	90,2%	56,7%	100,0%	75,0%
<b>Visit 3: Central Pavilion</b>							
Traveler	Non-Endorsed User Identifiers	N° Endorsements Collected	Endorsement Weights Sum	P2P Witnessing Strategy Confidence	Wi-Fi Scavenging Strategy Confidence	Combined Confidence Acquired	Confidence Threshold
User_5	{3, 4, 6, 7}	6	1,71	81,2%	52,9%	100,0%	75,0%
User_7	{}	5	1,45	68,8%	58,8%	100,0%	75,0%
User_6	{4}	5	1,45	69,0%	84,3%	100,0%	75,0%
User_3	{4}	5	1,47	70,0%	56,9%	100,0%	75,0%
User_2	{}	6	1,76	83,8%	82,4%	100,0%	75,0%
User_4	{}	2	0,60	28,6%	54,9%	83,5%	75,0%
User_1	{4}	6	1,79	85,0%	35,3%	100,0%	75,0%

The metrics collected on the route with 7 users are shown in Table 5.7. For this case we focus our analysis primarily on peer endorsement acquisition, as everything else is as expected

and what was explained above, for the 3 user experiment, also applies to this experiment. The growth of the endorsement weights sum is not as stable as above because: the amount of peer endorsements collected by each user is now different; the endorsement weights from some witnesses will be significantly higher at visit 2, as not all users issued endorsements to everyone at visit 1, and recall that endorsements weight is halved here after reusing witnesses on subsequent visits; in addition, due to the fact that not all users have issued endorsements for everyone, the weight of their endorsements is different, as a different number of good behavior reports are submitted to the reputation system for each of them, which makes their reputation differ (factor that also influences the lowering of the confidence threshold).

Now, delving deeper into the performance of the peer endorsement acquisition protocol in the extension of the route taken with 7 users, 2 anomalies occurred:

1. User\_5 was the one who stood out for the negative regarding the non-endorsement of their peers. Thanks to the instrumentalization of the app, we were able to assess that this was not due to the DB protocol failure, but to systematic errors in the establishment of incoming connections, which could be explained by the malfunction of the Bluetooth chip in the mobile device of the user in question, since in the others it worked just fine.
2. User\_4 failed to collect endorsements from the totality of the other users, even not considering User\_5. Through the instrumentalization of the app, we noticed that the claims were being rejected by the DB protocol, by the witnesses, as the prover (User\_4) was taking, on average, longer than acceptable time to respond to their challenges. This can be explained by realizing which mobile device the user in question was using: an Asus ZenFone 3 model released in 2016, equipped with a Qualcomm<sup>®</sup> Snapdragon<sup>™</sup> 625 CPU Octa-Core up to 2.0GHz, 4GiB of RAM, on top of Android 8 OS. The device is quite old (over 6 years old), it is running on an outdated Android OS and moreover it has poor specs for the current date.

Aside from these 2 anomalies, the peer endorsement acquisition protocol performed as expected, with all other users able to collect endorsements and endorse everyone else on every visit. That said, accounting for the failures experienced in these 2 anomalies, the peer endorsement acquisition success rate of a given peer was still quite high: 89,7%. Bearing this in mind, and the fact that all users were able to successfully validate their visits, we conclude that the P2P Witnessing strategy behaved quite well in a real-world setting.



### 5.2.2 Gamification Impact

At the end of the campus tour, we sent to the participants a Google form to assess the impact of the gamification elements (scoreboard, badges and in-app currency) on the user motivation to further use the P2P Witnessing strategy and the app itself.

**Gamification elements acquaintance** To immerse the user in the gamification elements developed, at the beginning of the tour we gave them a brief introduction about each one of them and the respective reward attribution mechanism, referring to the badges they could earn along the route. At the end of each visit, we encouraged users to access their wallets to check the number of gems they have earned and head over to the scoreboard to see where they stack up against the other participants. At the last point of interest of the route, we simulated the ticket clerks, requesting the price of the entrance ticket in cash or in gems (needless to say that everyone opted for gems), and therefore we scanned their app card, deducting the corresponding gems automatically.

**Participants background** The users were 22 and 23 years old and 6 out of 7 of them are used to visiting points of interest (such as viewpoints, museums and historical monuments), but none of them uses a platform or app to either enhance their tourist experience or get rewarded for the visits they make to points of interest.

**Engagement with gamification elements** Regarding the gamification elements, it was asked on a 5-point Likert scale, how engaged they felt (with 0 feeling disengaged and 5 feeling fully engaged), to which 5 out of 7 responded with a score greater than or equal to 4 (1 responded with a score of 2, another with a score of 3 and another with a score of 5, the rest responded with a score of 4).

**Preferred gamification elements** When asked which gamification element users liked the most or found most useful, the responses diverged, the users placed: **badges 1st** (4 out of 7), their justification was that they found it fun, knew what they were aiming for, and once they got their badges they felt an immediate sense of accomplishment; **scoreboard 2nd** (2 out of 7), their rationale was that they love to be well positioned on the scoreboard and enjoy the competition; **in-app currency 3rd** (1 out of 7), their justification was that it helps them save money corresponding to the entrance tickets cost.

**Gamification elements benefits** From the users point of view, the gamification elements can drive them to explore further because they are rewarded doing so. Furthermore it makes the experience more interactive and rewarding for something they would already be doing. It can also motivate them to further use the app because they are motivated to get badges to share with friends, climb up the scoreboard, and use the gems they have collected, which in doing so will earn more, creating the so-called *game loops*. An interesting factor that was assessed was that users also report that they feel more motivated to visit points of interest at the busiest hours, when there would be more tourists to issue more endorsements to and thus be better rewarded, a rather convenient factor for the best functioning of the P2P Witnessing strategy, even though it may inadvertently contribute to busier peak hours at attractions.

**Modifications to gamification elements** Several users suggested that we could develop an “*add friend*” feature, and create a private scoreboard composed exclusively of them (fostering the competition spirit focused only on the friends group). Furthermore, develop activities that could be carried out while they were together at points of interest and that they would be rewarded for doing so. Regarding the badges, users suggested that we could add some fun secret ones. Regarding the in-app currency, they suggested that we could extend its use to souvenir shops.

**User adherence to the P2P Witnessing strategy** When the users were asked if they would voluntarily enable the P2P Witnessing strategy, all answered yes (fulfilling **G7**), as it improves their results not only in the chances of seeing their visit successfully validated, but also allows them to be additionally rewarded for something they would already be doing, and the downside is almost negligible. They also acknowledged that the use of the P2P Witnessing strategy (with the provision of additional rewards) can be essential for those who want to: compete for the first places on the scoreboard; maximize their gem earnings to save on future visits; and get badges for being a good comrade endorsing peer visits.

**User adherence to the app** When users were asked if they would use the app again for future visits, 6 out of 7 users answered yes (meeting **G8**), because of the incentive to be rewarded for something they would already be doing and the possibility to save on future visit tickets, in addition, they also highlighted the ease of use and simplicity of the UI. An interesting justification for an affirmative answer was: “*because now I have tons of gems to spend*” (which the user collected on the route traveled) corroborating the creation of the game loop already foreseen.

However, most also say they want to see more features developed in the context of making the visits to points of interest more dynamic (such as on-site challenges).

### 5.3 Summary

In the evaluation of this work in laboratory experiments, we concluded that: the P2P Witnessing strategy is resilient against systematic prover-witness collusion, the peer endorsement validation is responsive thus providing real-time user feedback, the strategy is resilient against prover-prover collusion by measuring the DB protocol acceptance rate for honest provers versus colluding provers, the peer endorsement acquisition protocol is efficient thus not requiring users to stay close to each other for a long time, and finally, the impact of the new strategy on the mobile device battery is not a factor of concern for the user experience.

In addition, the evaluation with users allowed us to conclude that: the P2P Witnessing strategy is feasible in a real-world scenario, and the built-in gamification elements served their purpose, encouraging users to further use the app and adopt the new strategy.

Overall, the evaluation demonstrated that the objectives of this work were successfully achieved, as it was possible to verify that all the solution guidelines defined in Section 4.2 were fulfilled.



## Chapter 6

# Conclusion

This work successfully delivered a new version of CROSS City extended with a new peer-based collusion-resilient location certification strategy (P2P Witnessing) that addresses the weaknesses of the previous infrastructure-based strategies, providing several advantages to the system, including: extension of its operation to places with no surrounding infrastructure, ability to validate user visits in real-time with temporal granularity, and fortification against location evidence transfer between users. In addition, we also integrated gamification elements (scoreboard, badges and in-app currency) into the CROSS City app that reward users for visiting points of interest but also for using the P2P Witnessing strategy. The evaluation demonstrated that they have the potential to engage and encourage users not only to use the app, but also to adopt this strategy, which is particularly important, as it relies on user participation.

Moreover, we covered different types of location certification techniques, both infrastructure-based and peer-based, identifying their properties, strengths and weaknesses. We saw different strategies to improve the user experience (namely, based on augmented reality) and how to adapt the UI to the heterogeneity of devices and users. We also verified some of the potential benefits of gamification (namely, user engagement and encouragement), which gamification features to adopt and how different motivational elements can be used in gamification.

### 6.1 Achievements

The achievements of this work were the following:

- We explored background and related work in the areas of location certification techniques, user experience and gamification;

- We completely re-implemented the CROSS City prototype system, improving the code base and extending its functionality to offer a better UI;
- We introduced and integrated into CROSS City a new peer-based collusion-resilient location certification strategy, named P2P Witnessing;
- We developed new extrinsic gamification elements into the CROSS City app;
- We carried out a practical evaluation covering laboratory experiments and test with users in a real-world scenario (a university campus tour) that demonstrated that this work was a success, since it verified the fulfillment of all the pre-established solution guidelines.
- Finally, we articulated this work with two others: *CROSS City Cloud* [MP22a] that enables the processing and storage of CROSS City data in a cloud infrastructure; as well as *SureRepute* [MP22b] which manages the reputation of the app users.

## 6.2 Future Work

Regarding the P2P Witnessing strategy, one can explore the possibility of implementing it in a layer below the application layer, to obtain a more accurate measurement of the DB challenge response times, sacrificing, however, its extensibility and ease of installation on the user device, in favor of a stricter location certification.

We thought of many other gamification and user experience features, but due to time constraints, we only developed the most important ones that best suited our goal, which was to encourage users to use P2P Witnessing strategy. We present here the other features that we have designed (some of the screen mockups are illustrated in Appendix B).

It is important to balance gamification features that encompass both the extrinsic motivational elements (like the ones we developed), which aim to provide users with a sense of accomplishment and yearning for more experiences, and the intrinsic ones, which concerns the pure pleasure of doing the activity. Thus, we list below the gamification features that can address the intrinsic motivation of the user:

- *Challenges*: When visitors stop at a specific point of interest while following a tourist route, provide challenges (for example: multiple choice questions, correct image identification, and marker hunt) that can help the user to better enjoy their visit and discover interesting things that the place has available. By getting the user to spend more time onsite, location certification strategies will work better;
- *Likes and Recommendations*: An indirect means of communication from which users can

get ideas about which tourist routes to take. This would work through a system of likes, where users could like the routes they have traveled and, in turn, they would be recommended routes according to the likes of other users who tend to like the same routes.

- *Social Media Sharing*: Allow users to share badge earning and route completion certificates on their social networks.

In addition, we also designed other user experience features, which we list below:

- *UI Customization*: The user would be given the ability to customize the UI, such as changing the color scheme to one that best suits them or increasing the font size if the user has eyestrain;
- *Pin Routes*: Allow users to pin routes they may want to follow later to access them faster and stay organized;
- *Route Planning Assistance*: Users could get a planning containing the order in which the route points of interest should be visited to minimize the distance they need to travel. Furthermore, users could click on points of interest on the map and it would be shown how far away they are from them.
- *Background Music*: During the visit, the app would play music to make the experience more relaxing;
- *Context Awareness*: Automatically switch mobile phone sound mode to silent when the user enters a quiet place, such as a museum or church;
- *PoI Proximity Notifications*: Notify users when they are close to a point of interest suggesting that they follow the route it is a part of. Notifications could be turned off by users if they do not want to be disturbed;
- *User-Friendly Authentication*: Users would have the option to sign up and log in with Google. Since most Android users must have a Google account set up on their mobile device, they will be able to authenticate with just one tap, making the authentication ceremony less tedious.

In addition to extending CROSS City with these additional gamification and user experience features, this system also has the potential to be extended beyond the tourism domain, such as in malls whose points of interest would be stores and gems would serve as in-store discount vouchers, for example, as well as for package deliveries, parking, etc.

With this work, we have shown how location certificates can go beyond a security credential and become part of engaging consumer applications.





# Bibliography

- [AA20] Hosam Alamlah and Ali Abdullah S AlQahtani. A cheat-proof system to validate gps location data. In *International Conference on Electro Information Technology (EIT)*, pages 190–193. IEEE, 2020.
- [AT03] Leena Arhippainen and Marika Tähti. Empirical evaluation of user experience in two adaptive mobile application prototypes. In *MUM. Proceedings of the 2nd International Conference on Mobile and Ubiquitous Multimedia*, number 011, pages 27–34. Citeseer, 2003.
- [BGJG13] Gabriel Barata, Sandra Gama, Joaquim Jorge, and Daniel Gonçalves. Engaging engineering students with gamification. In *5th International Conference on Games and Virtual Worlds for Serious Applications (VS-Games)*, pages 1–8. IEEE, 2013.
- [CCCDP13] Eyüp S Canlar, Mauro Conti, Bruno Crispo, and Roberto Di Pietro. CREPUS-COLO: A collusion resistant privacy preserving location verification system. In *International Conference on Risks and Security of Internet and Systems (CRiSIS)*, pages 1–9. IEEE, 2013.
- [CEP22] Rui L Claro, Samih Eisa, and Miguel L Pardal. CROSS scavenger: Wi-Fi signal collection for time-bound location proofs. 2022. To be submitted to IEEE Access.
- [DL18] Amir Dirin and Teemu H Laine. User experience in mobile augmented reality: Emotions, challenges, opportunities and best practices. *Computers*, 7(2):33, 2018.
- [FP18] Joao Ferreira and Miguel L Pardal. Witness-based location proofs for mobile devices. In *17th International Symposium on Network Computing and Applications (NCA)*, pages 1–4. IEEE, 2018.
- [Fra21] Miguel Cordeiro Francisco. SurePresence: Location Proofs for Wearable and Kiosk Devices. Master’s thesis, Instituto Superior Técnico, Universidade de Lisboa, Portugal, 2021.

- [HHB<sup>+</sup>18] Jamil Hussain, Anees Ul Hassan, Hafiz Syed Muhammad Bilal, Rahman Ali, Muhammad Afzal, Shujaat Hussain, Jaehun Bang, Oresti Banos, and Sungyoung Lee. Model-based adaptive user interface based on context and user experience evaluation. *Journal on Multimodal User Interfaces*, 12(1):1–16, 2018.
- [HKS14] Juho Hamari, Jonna Koivisto, and Harri Sarsa. Does gamification work?—a literature review of empirical studies on gamification. In *47th Hawaii international conference on system sciences*, pages 3025–3034. IEEE, 2014.
- [Lik32] Rensis Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.
- [MCP20] Gabriel A Maia, Rui L Claro, and Miguel L Pardal. CROSS City: Wi-Fi Location Proofs for Smart Tourism. In *International Conference on Ad-Hoc Networks and Wireless*, pages 241–253. Springer, 2020.
- [MP22a] Lucas Vicente Miguel Pardal, Samih Eisa. CROSS City Cloud: Extension, Deployment and Operation. Master’s thesis, Instituto Superior Técnico, Universidade de Lisboa, Portugal, 2022.
- [MP22b] Rafael Figueiredo Miguel Pardal, Samih Eisa. SureRepute: User Reputation in Location Certification Systems. Master’s thesis, Instituto Superior Técnico, Universidade de Lisboa, Portugal, 2022.
- [NSY<sup>+</sup>20] Mohammad Reza Nosouhi, Keshav Sood, Shui Yu, Marthie Grobler, and Jingwen Zhang. PASPORT: A Secure and Private Location Proof Generation and Verification Framework. *Transactions on Computational Social Systems*, 7(2):293–307, 2020.
- [TCB10] Manoop Talasila, Reza Curtmola, and Cristian Borcea. LINK: Location verification through immediate neighbors knowledge. In *International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pages 210–223. Springer, 2010.
- [TF17] Sarah-Kristin Thiel and Peter Fröhlich. Gamification as motivation to engage in location-based public participation? In *Progress in location-based services*, pages 399–421. Springer, 2017.
- [UTAY14] Yoshitaka Ueyama, Morihiko Tamai, Yutaka Arakawa, and Keiichi Yasumoto. Gamification-based incentive mechanism for participatory sensing. In *Interna-*

*tional Conference on Pervasive Computing and Communication Workshops (Per-Com Workshops)*, pages 98–103. IEEE, 2014.

[XWB13] Feifei Xu, Jessika Weber, and Dimitrios Buhalis. Gamification in tourism. In *Information and communication technologies in tourism*, pages 525–537. Springer, 2013.

[ZC11] Zhichao Zhu and Guohong Cao. APPLAUS: A privacy-preserving location proof updating system for location-based services. In *Proceedings INFOCOM*, pages 1889–1897. IEEE, 2011.



## Appendix A

# CROSS City Mobile Application Screens

In this Appendix we present some of the screens of the CROSS City mobile application we developed during this dissertation. We also provide a [demo video](#)<sup>4</sup> of the app.

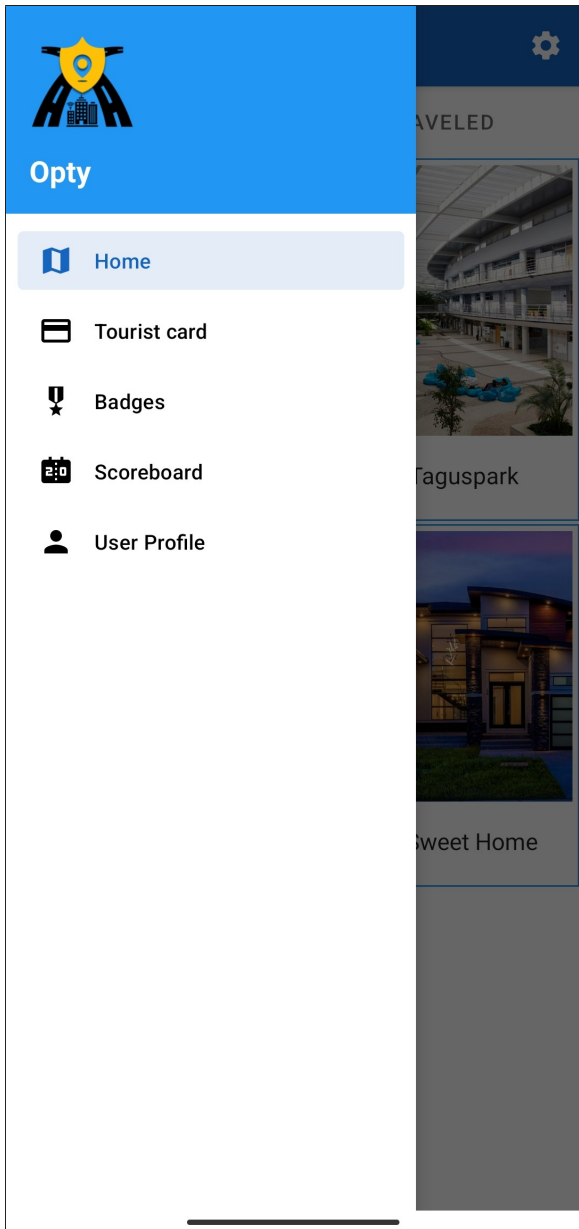


Figure A.1: Navigation drawer with app menus.

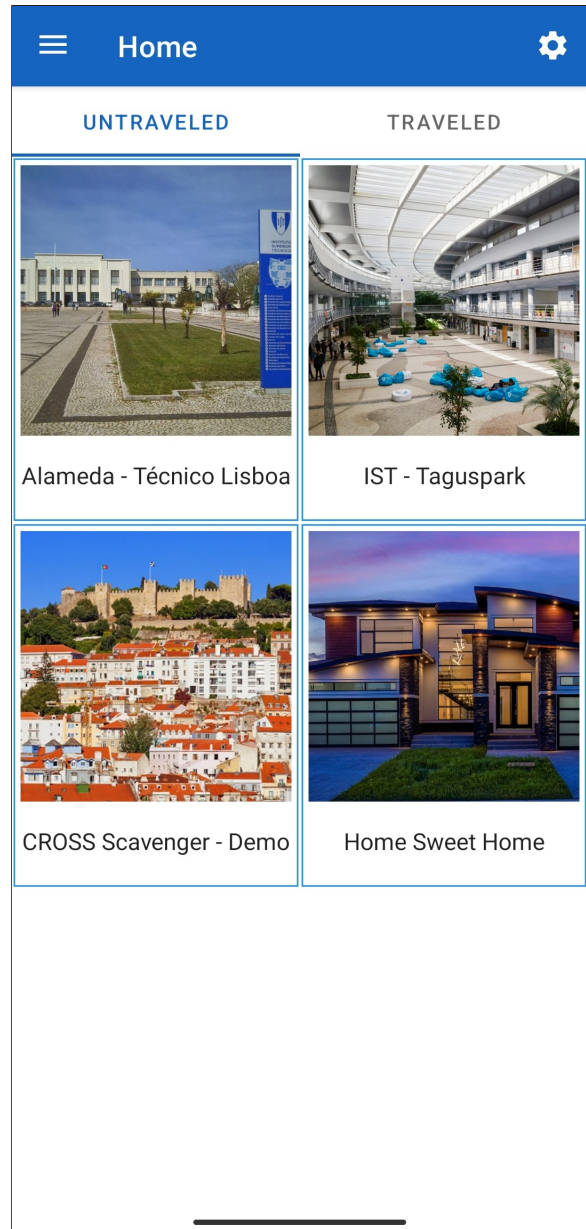


Figure A.2: List of untraveled routes.

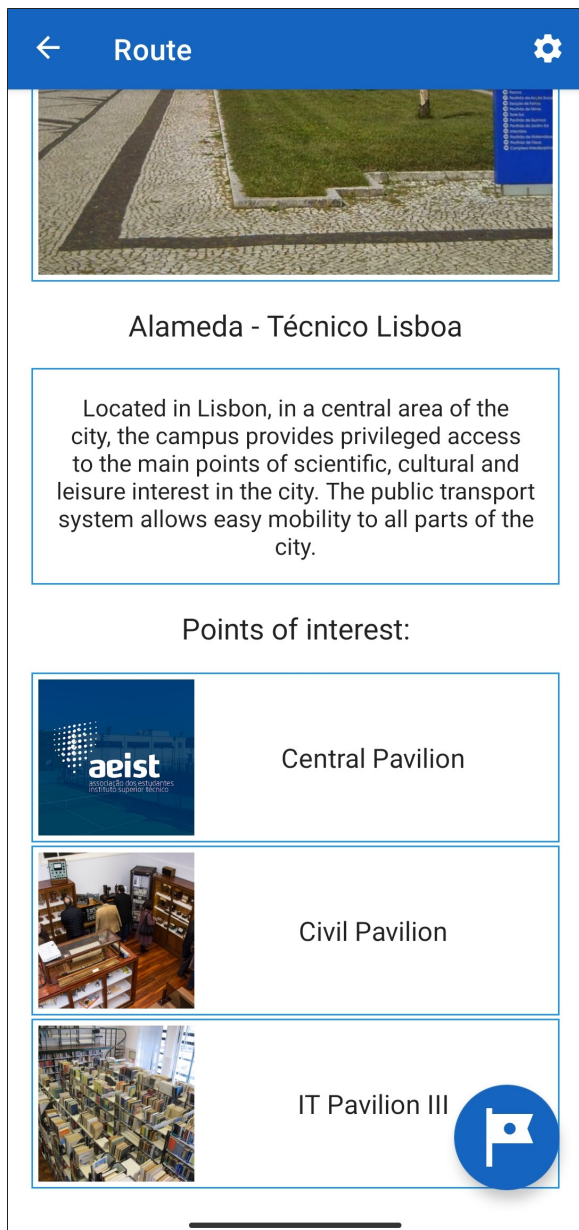


Figure A.3: Route yet untraveled. Typing on each of the points of interest navigates you to the respective screen in Figure A.4. Clicking on the start flag initiates the route, navigating you to the screen in Figure A.6.

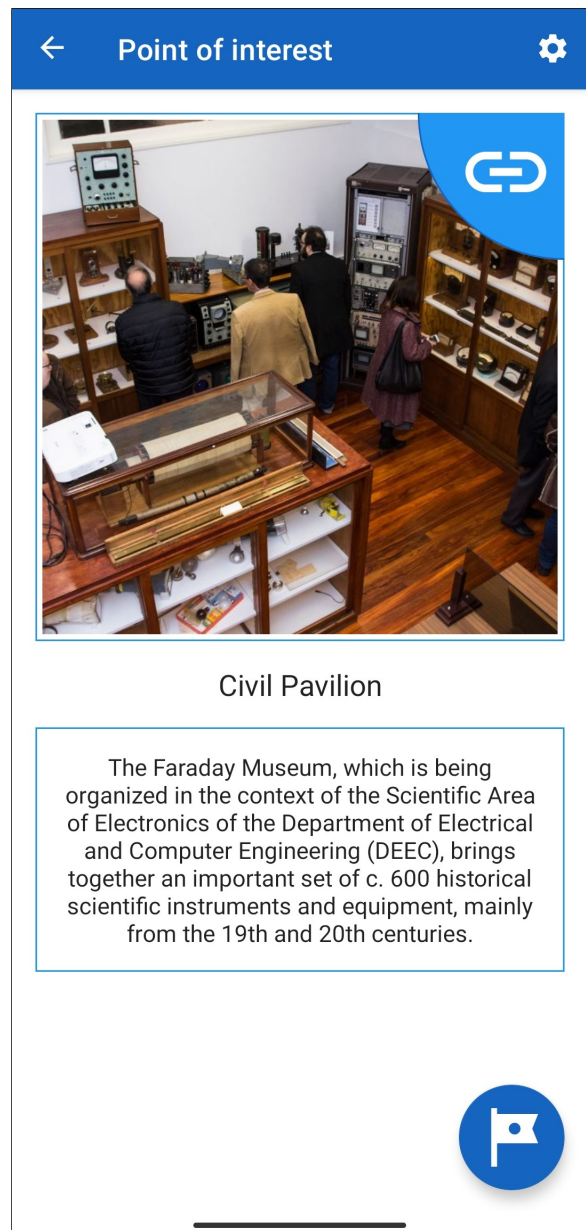


Figure A.4: Point of interest yet untraveled. Clicking the link button in the upper right corner of the image takes you to the point-of-interest website. Clicking on the start flag initiates the visit, navigating to the screen in Figure A.7.

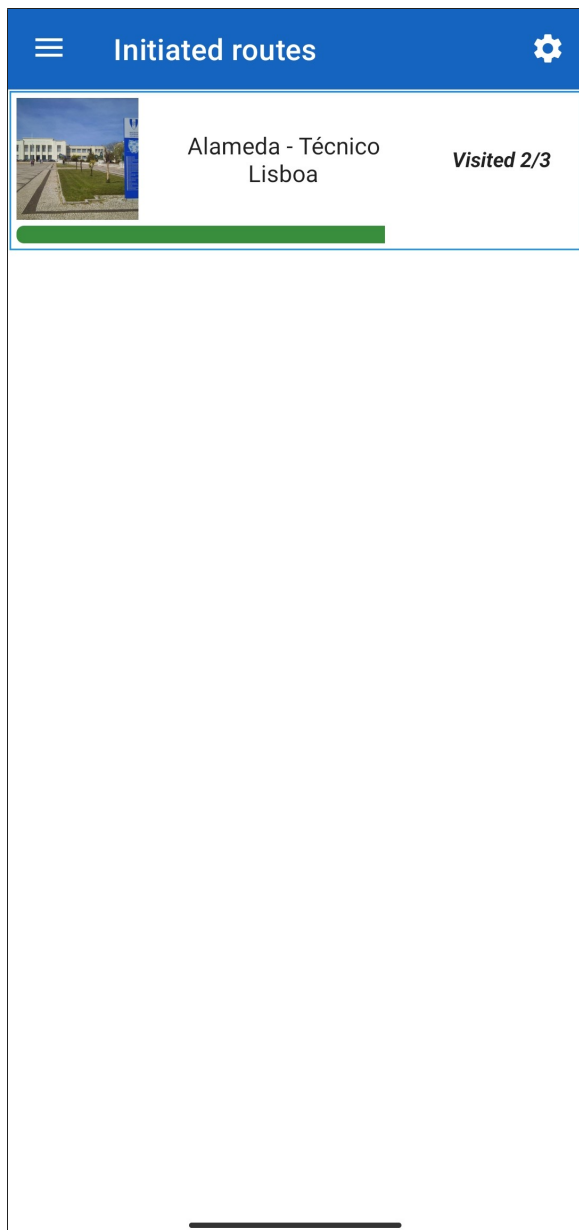


Figure A.5: List of routes already initiated. All routes already initiated are displayed here. The green bar indicates the progress on the route. Typing in one of these routes takes you to the corresponding screen in Figure A.6.

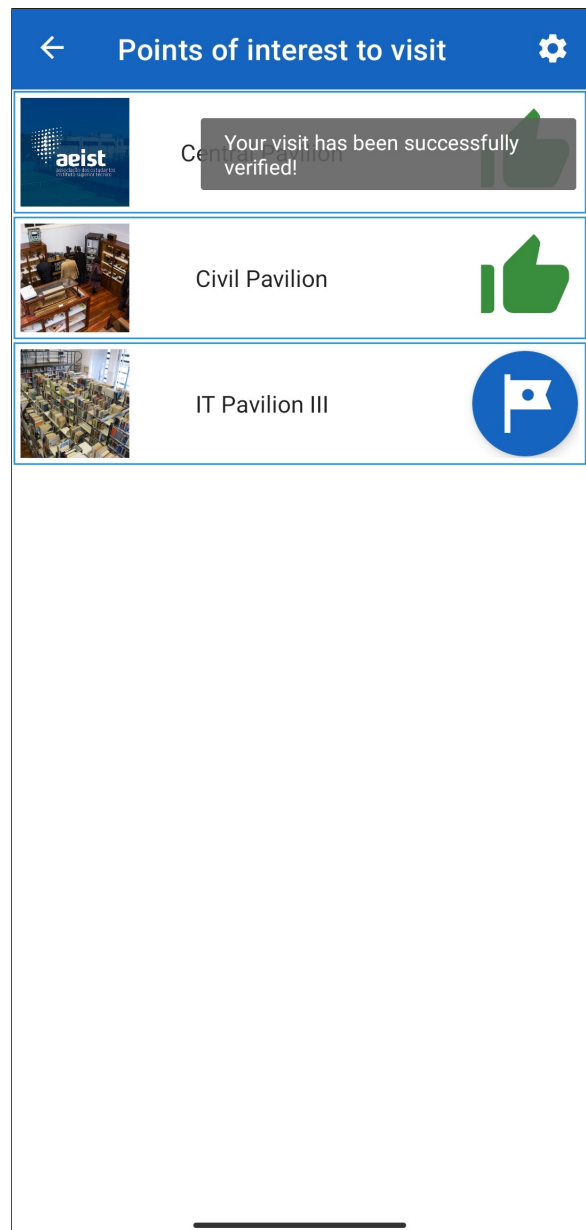


Figure A.6: Route already initiated. Here, the visit statuses of the points of interest that make up the route are displayed, if different from verified, a start flag is displayed that can be clicked to start the corresponding visit, navigating to the screen in Figure A.7.



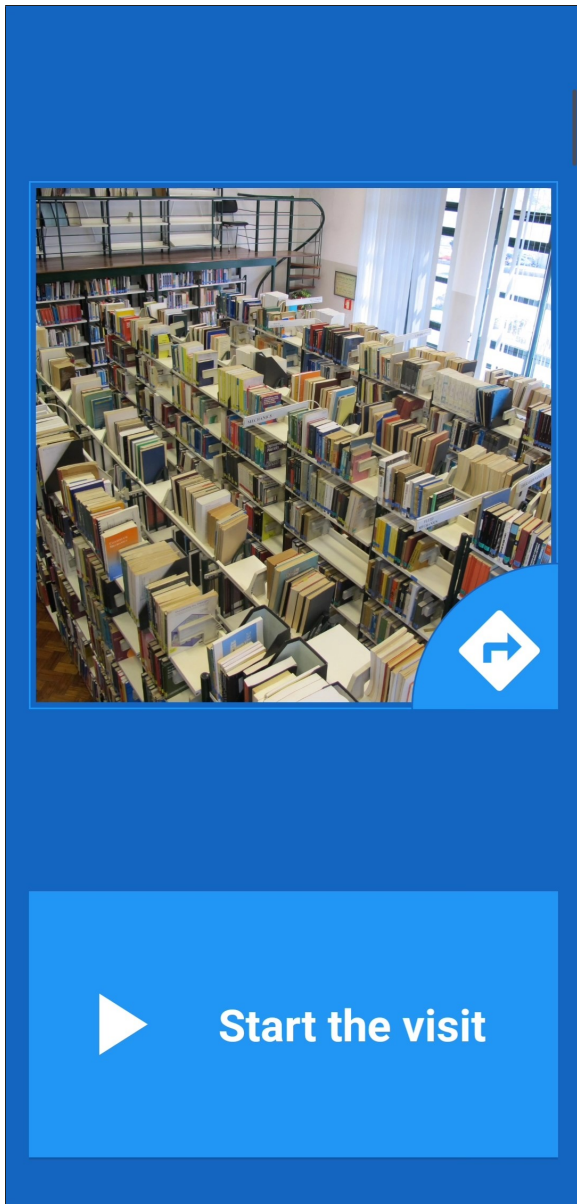


Figure A.7: Start of visit.

Clicking the directions button takes you to Google Maps, showing the directions from your current location and heading to the point of interest in question. Clicking the visit start button begins the collection of visit evidence, navigating to the screen in Figure A.8.

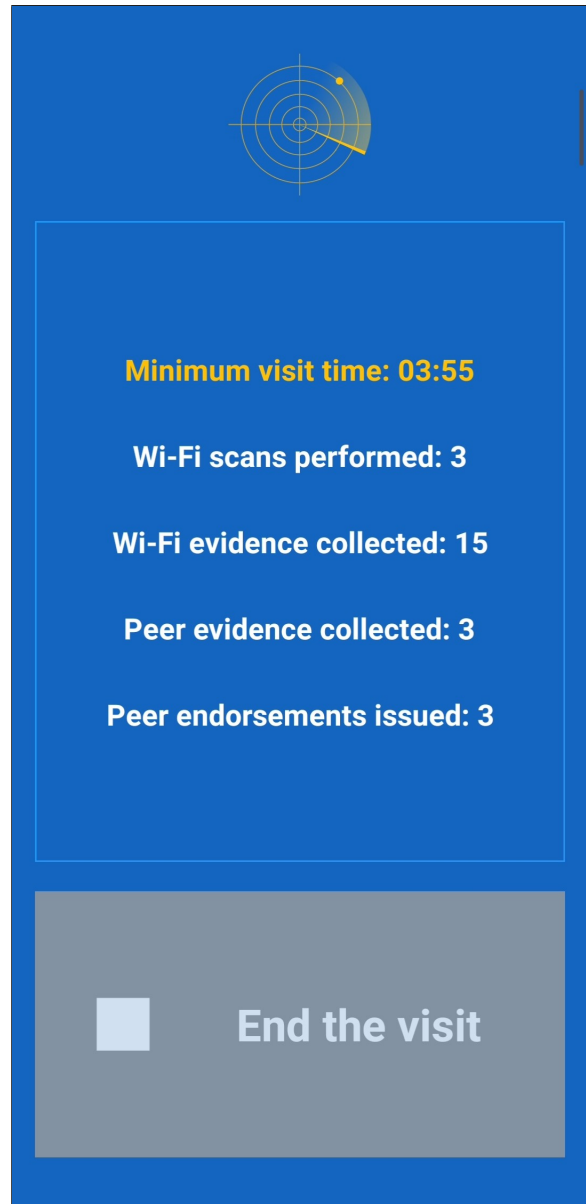


Figure A.8: Feedback on evidence collected during the visit.

Here, the feedback from the collected visit evidence is shown. The minimum visit time is a countdown, until the user can submit the visit (until then the end visit button appears disabled), if the visit is successfully verified, it navigates you to the screen in Figure A.9.

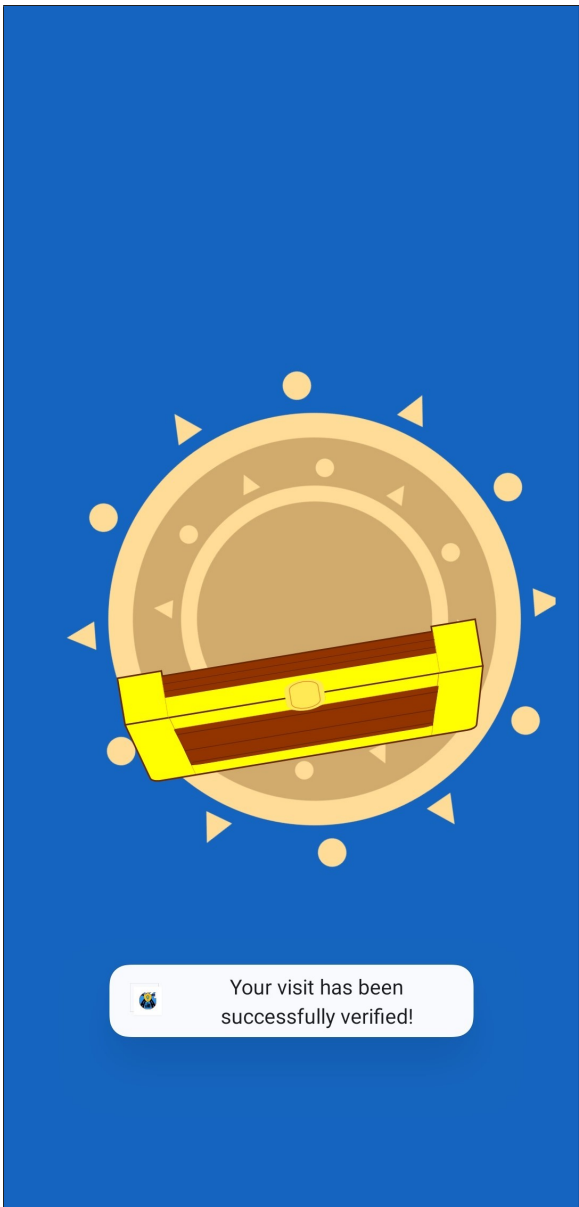


Figure A.9: Opening visit rewards chest.

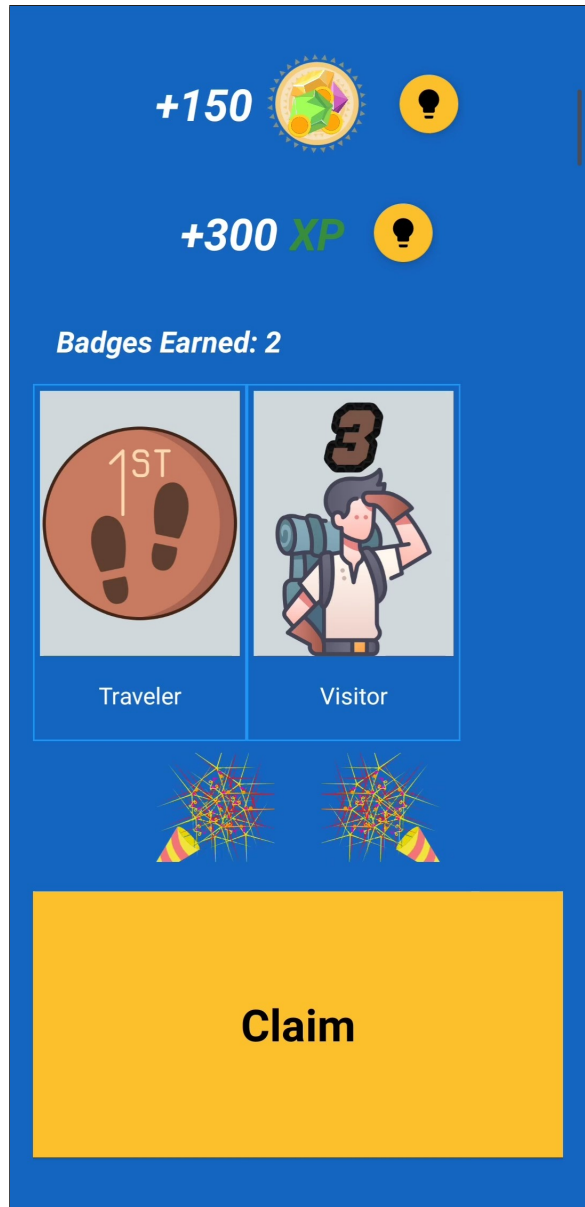


Figure A.10: Visit rewards.  
Visit rewards received upon visit validation.

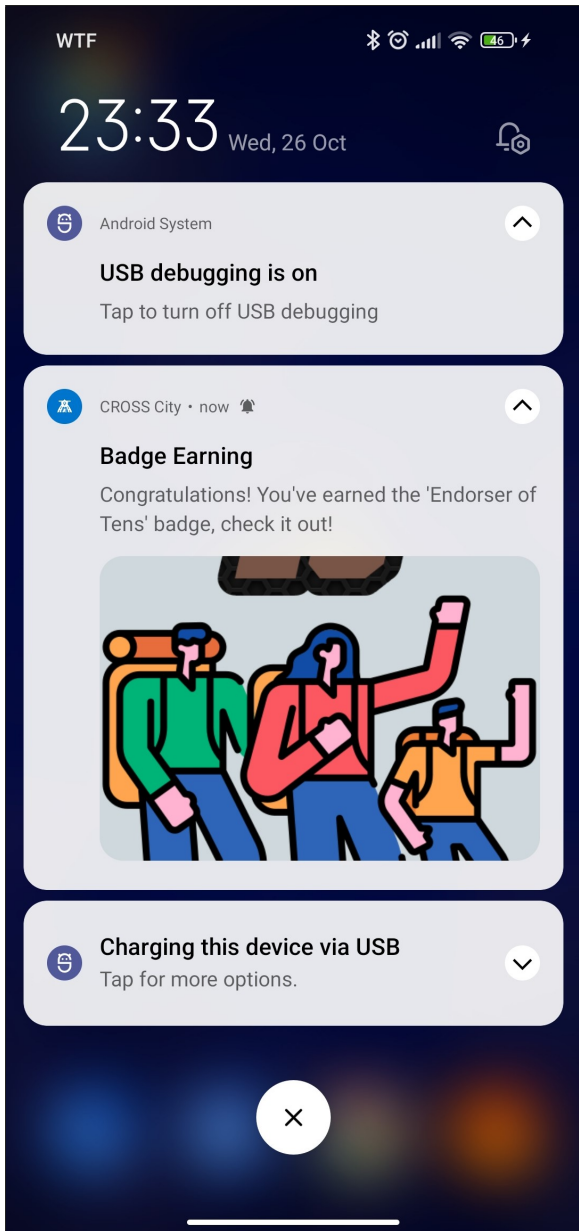


Figure A.11: Rewards notification.

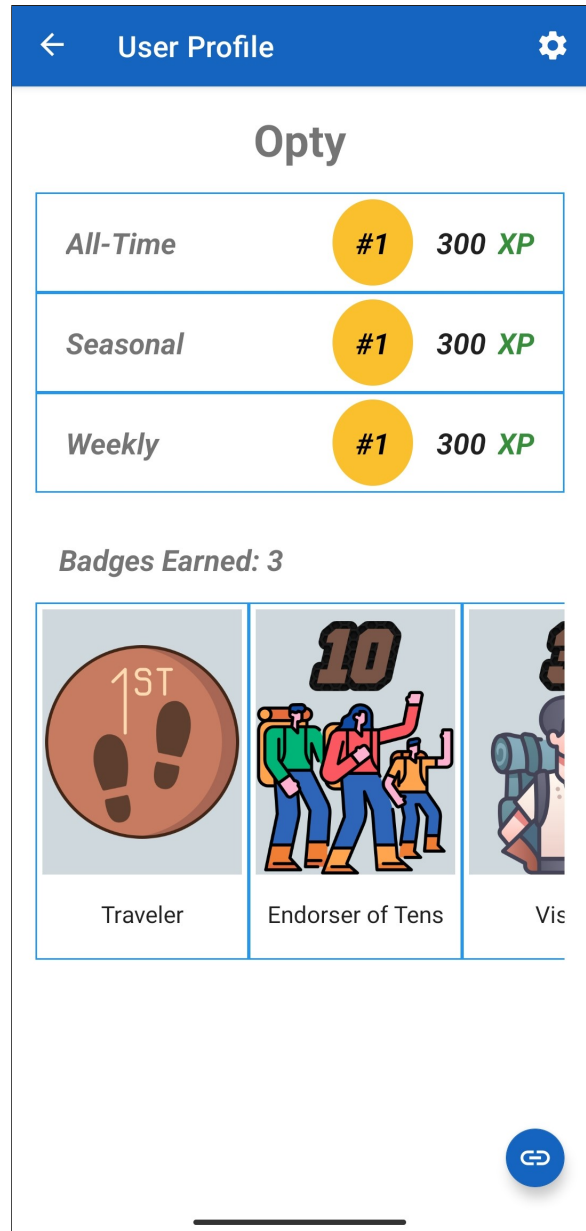


Figure A.12: User profile.

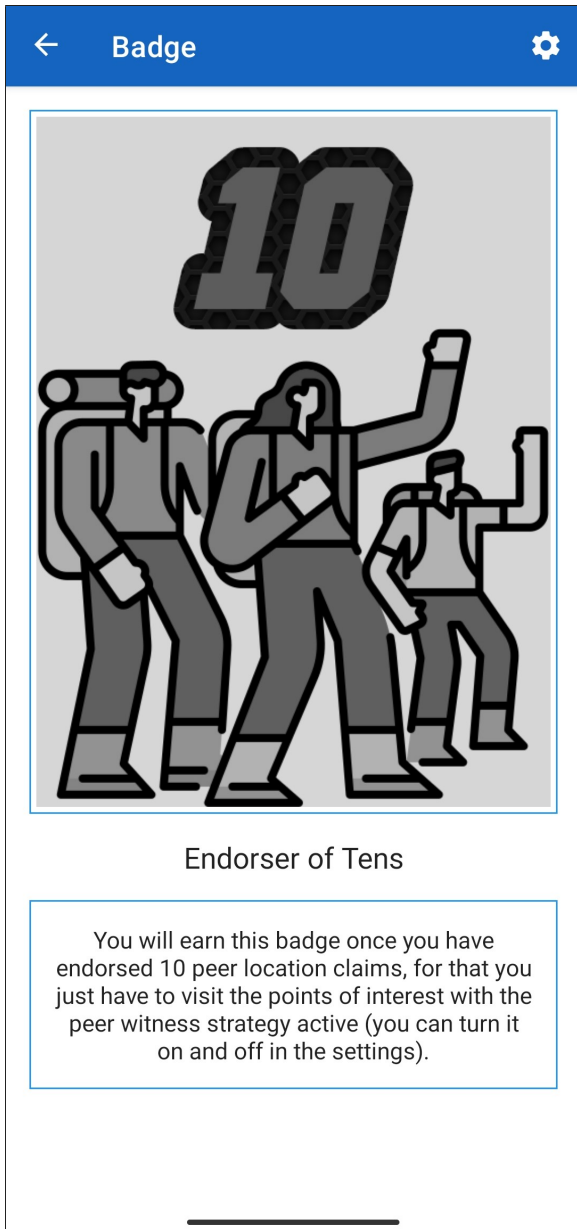


Figure A.13: Badge yet unearned.

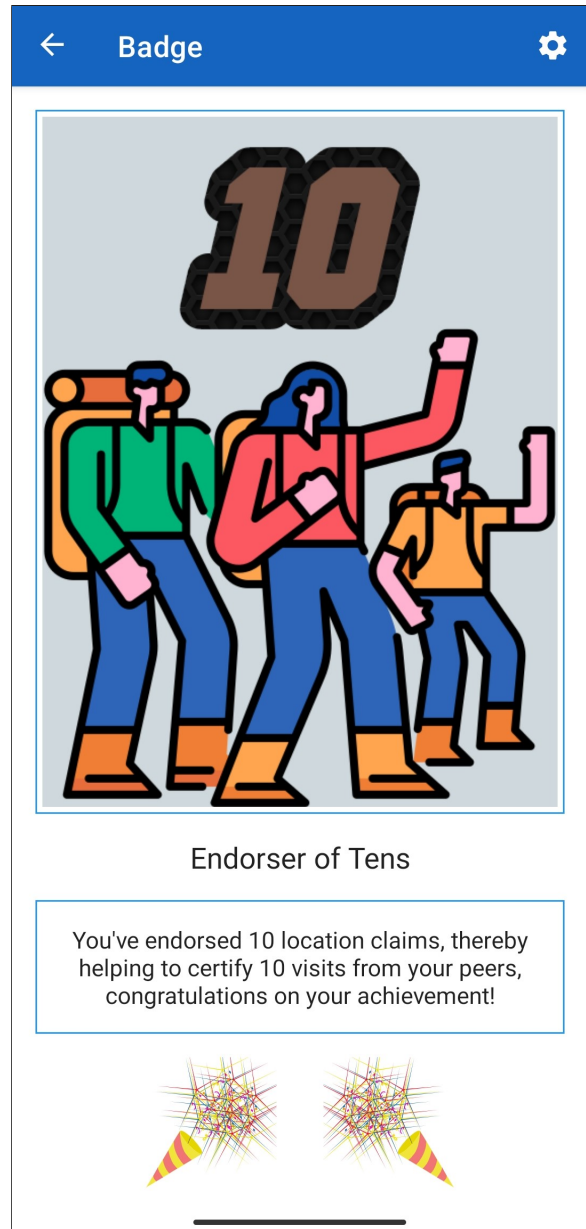


Figure A.14: Badge earned.

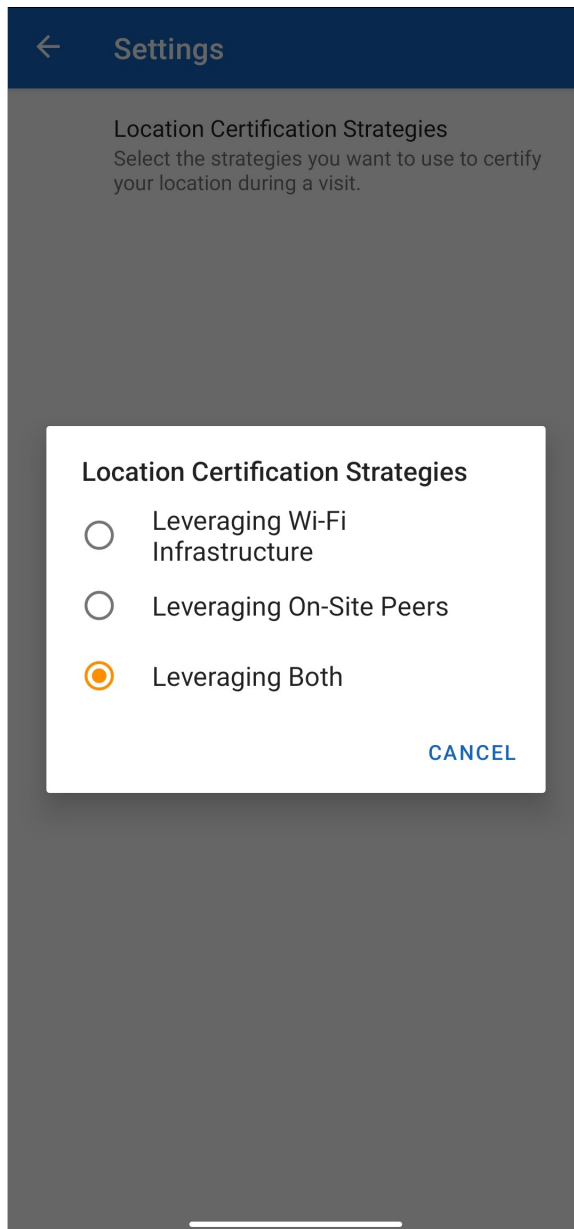


Figure A.15: Choice of location certification strategies scheme to be used in a visit.



## Appendix B

# CROSS City Future Extensions

In this Appendix we present some of the screen mockups (developed in the Justinmind <sup>1</sup> tool) of possible future extensions to the CROSS City system proposed in Section 6.2.

---

<sup>1</sup><https://www.justinmind.com/>

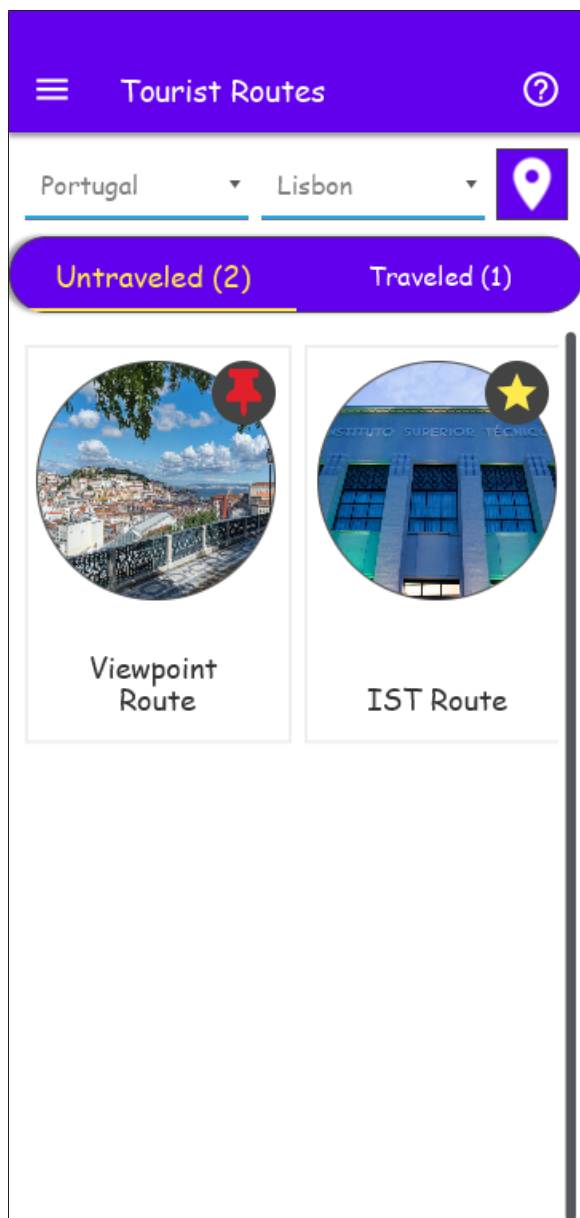


Figure B.1: List of untraveled routes. The routes marked with thumbtacks are those that the user has pinned and those with stars are those recommended by the system to the user.

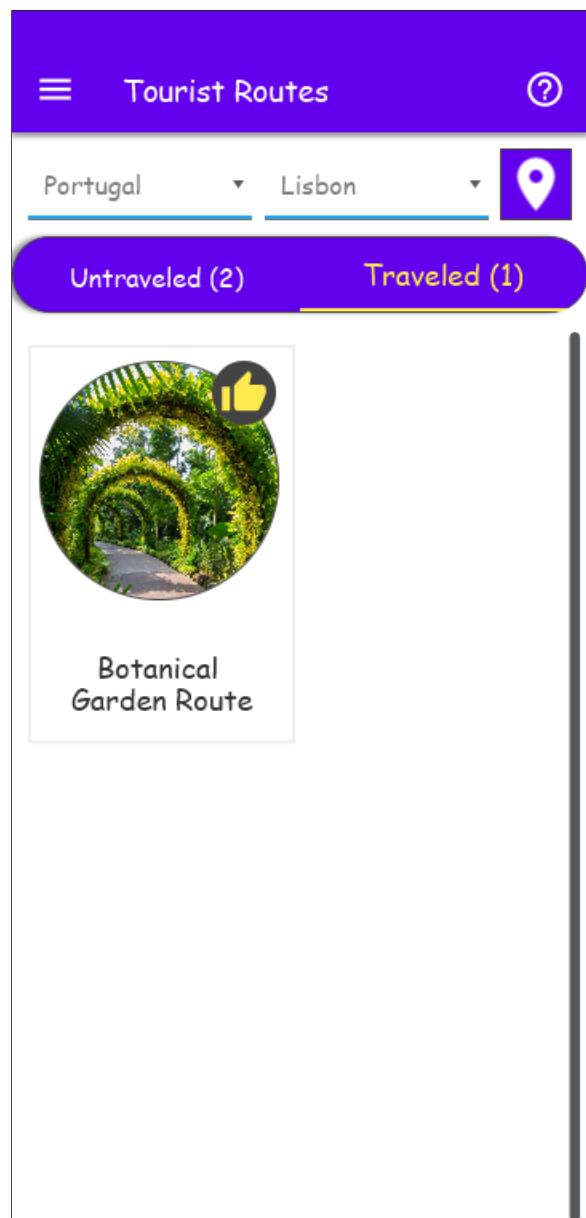


Figure B.2: List of traveled routes. The routes marked with thumbs up are the ones the user liked.



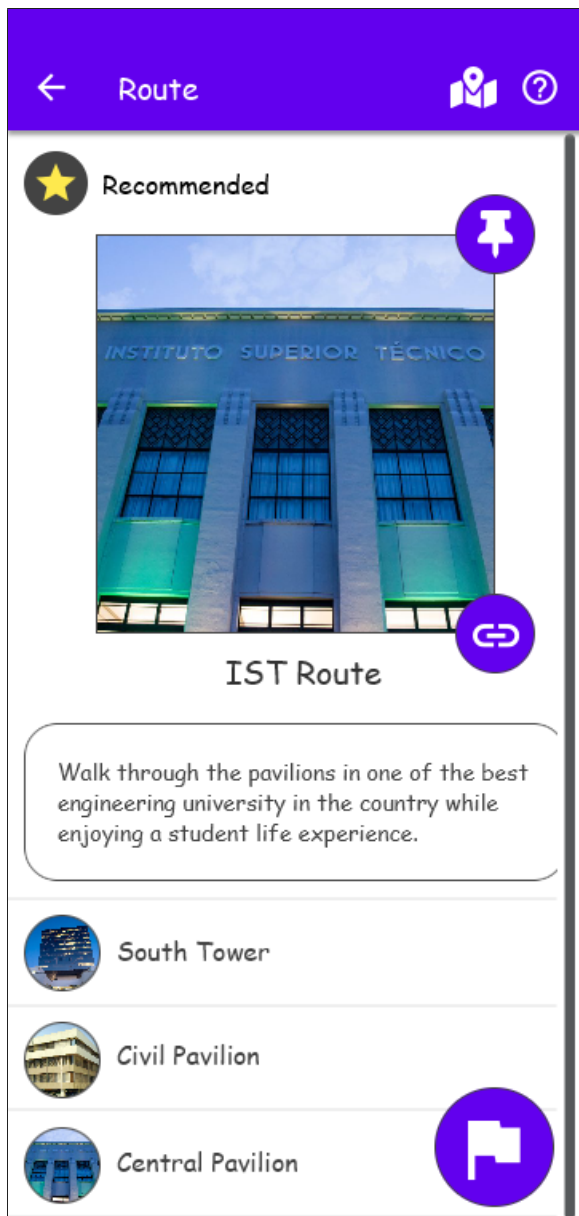


Figure B.3: Route yet untraveled.

Here the user would be able to pin the route, open the map of points of interest and initiate the route.

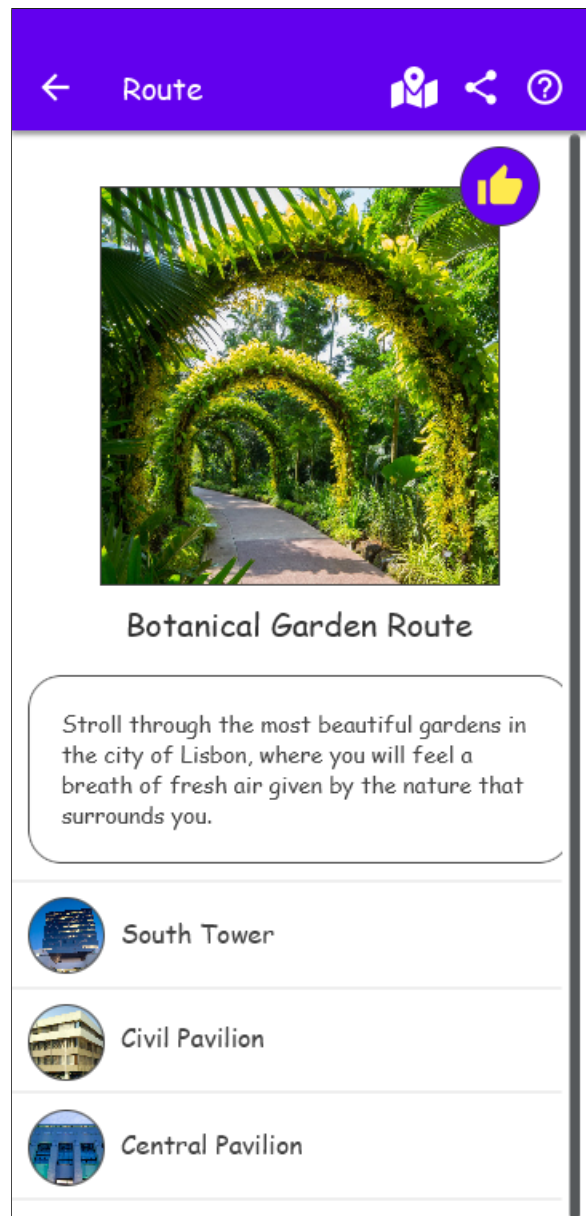


Figure B.4: Route already traveled.

Here the user would be able to like the route, open the map of points of interest and share the route completion certificate on their social networks.

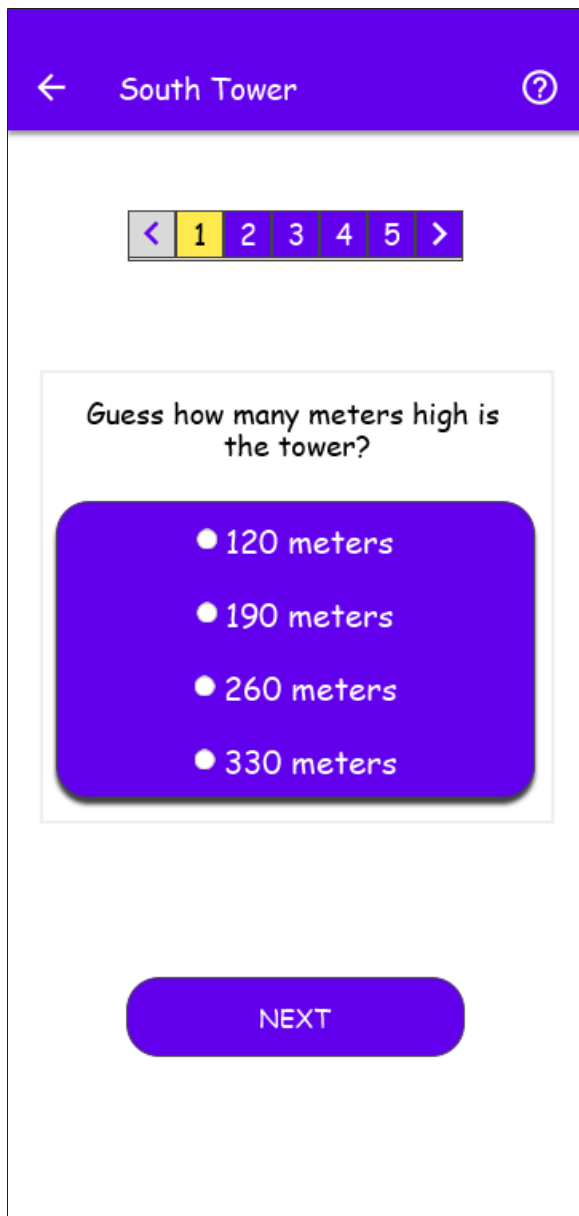


Figure B.5: Multiple choice challenge. This is one of the multiple choice challenges the user could face while at a point of interest; the challenges could be scrolled through the top bar.

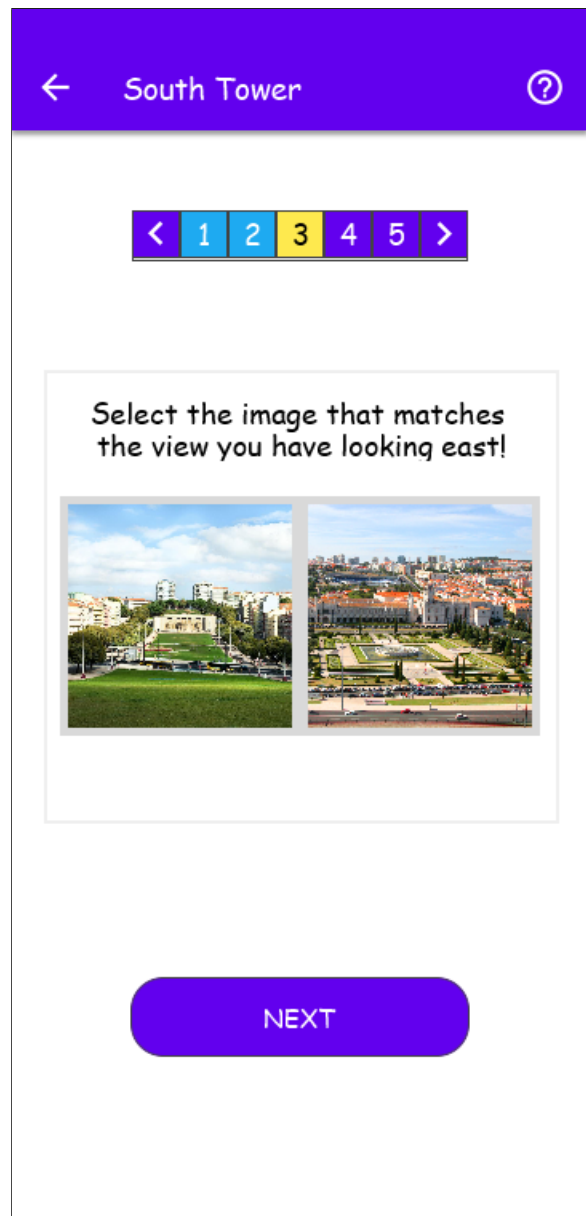


Figure B.6: Challenge of identifying the correct image. In this challenge, the user would have to identify the correct image, which is, for example, a landscape that can be observed at a certain location of the point of interest.

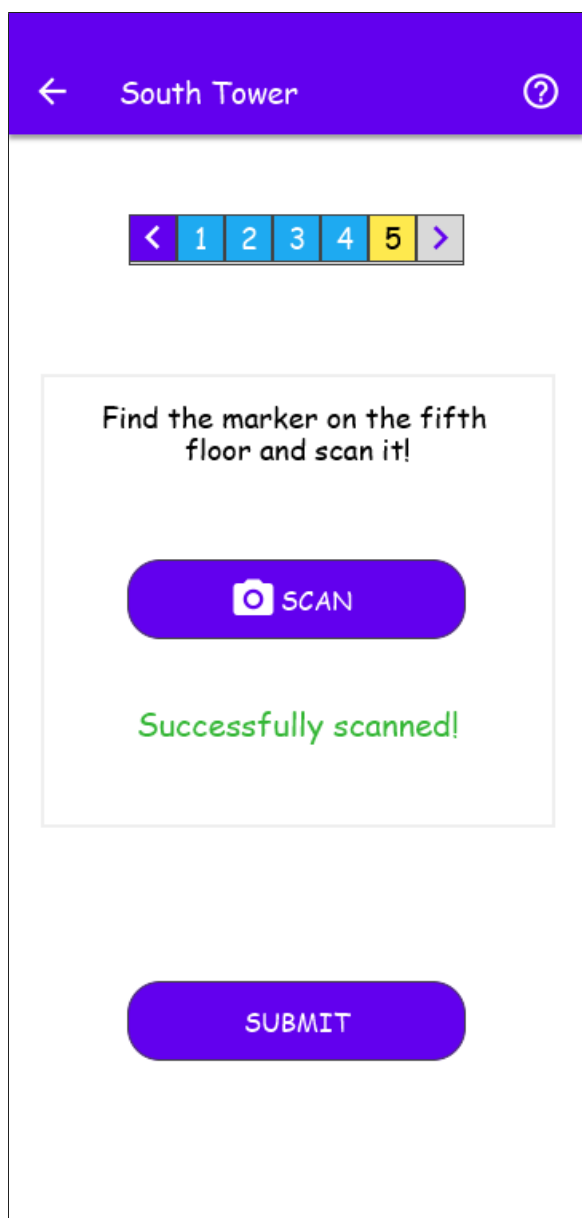


Figure B.7: Marker hunt challenge. In this challenge, users would need to go looking for a marker that is somewhere in the point of interest and scan it with their mobile device camera.

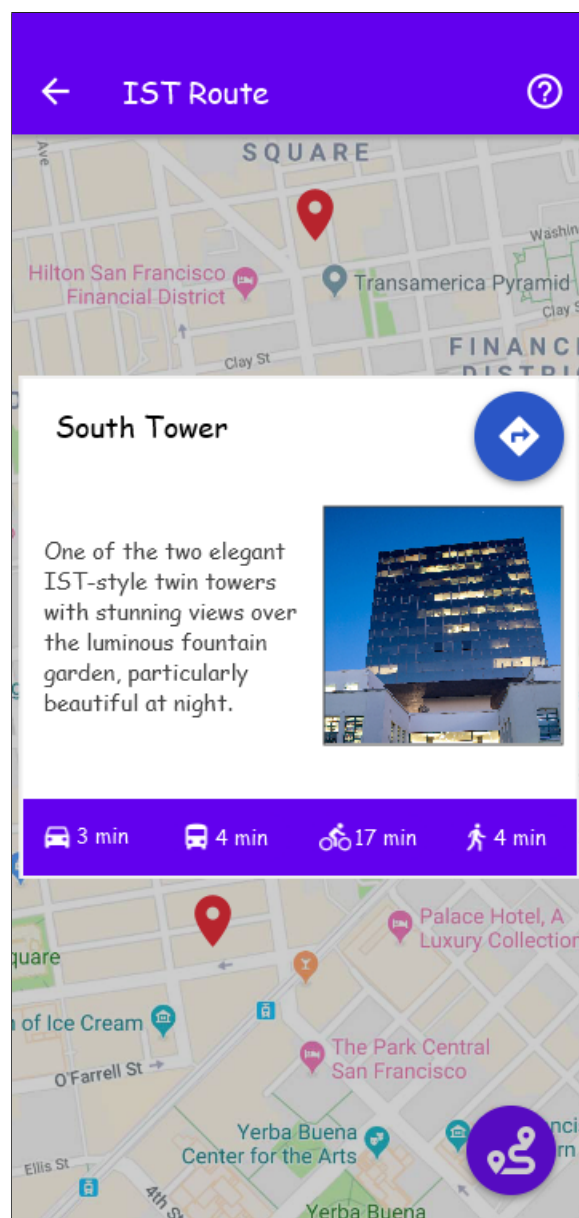


Figure B.8: How far is the user from the point of interest. Here the user could see how far from the selected point of interest on the map they are, taking into account different ways of moving. In addition, the user could get directions on Google Maps to navigate the route and see which is the shortest way to go.

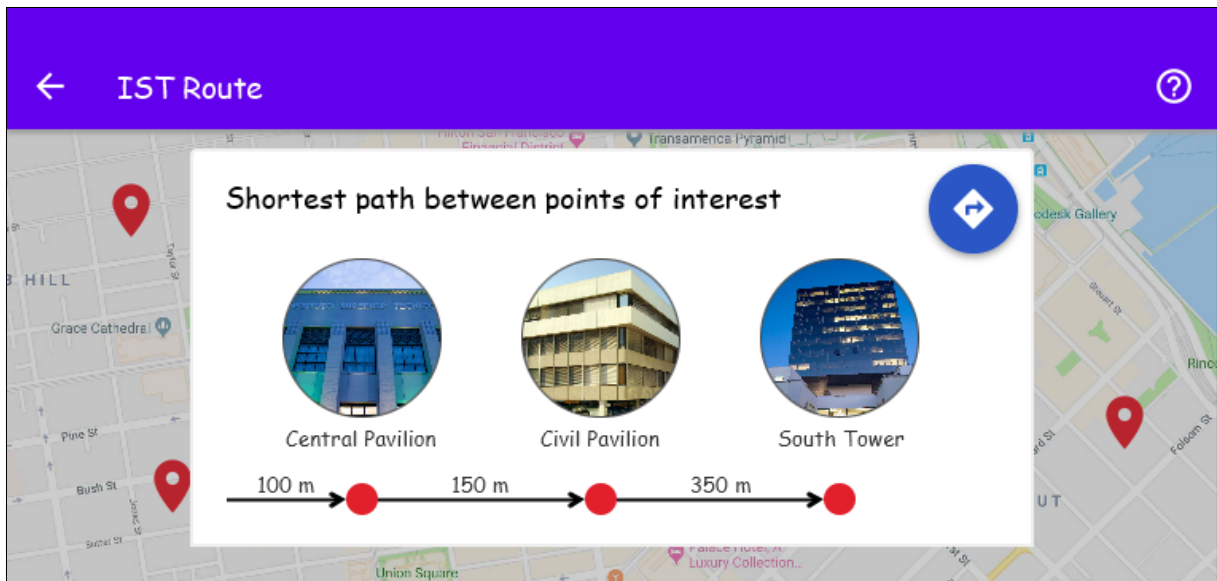


Figure B.9: Shortest path of a route.

Here, the ideal route for the user to follow to cover a shorter distance is shown.

## Appendix C

# CROSS City Campus Tour - Google Forms

This was the form given to users who participated in the campus tour, the results were analyzed in Section 5.2.2.

# CROSS City Campus Tour

This form aims to collect information related to the tour of the Alameda campus, namely the impact of gamification elements on user motivation.

**\*Obrigatório**

1. How old are you? \*

---

2. Do you often visit points of interest (e.g. historic monuments, museums or viewpoints)? \*

*Marcar apenas uma oval.*

Yes

No

3. Do you usually use some platform or app to either enhance your tourist experience or get rewarded for the visits you make to points of interest? \*

*Marcar apenas uma oval.*

Yes

No

4. Did you feel engaged with the gamification elements (scoreboard, badges and app currency) that are integrated into the app? \*

*Marcar apenas uma oval.*

1      2      3      4      5

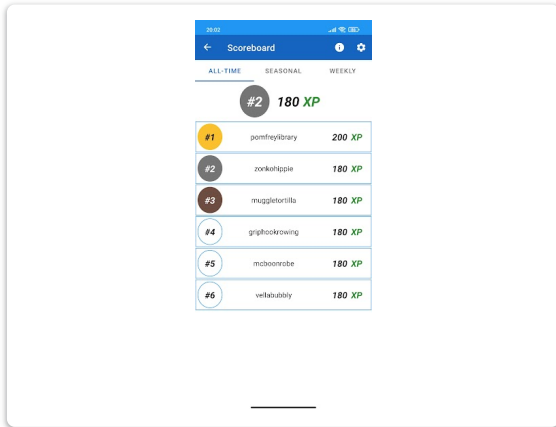
---

Not at all                  Completely

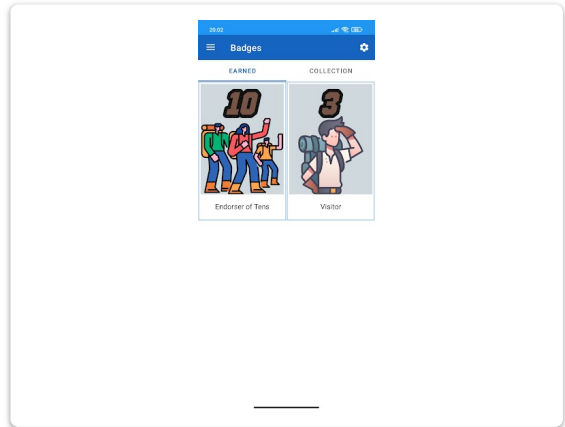
---

5. Which gamification element did you like the most or did you find most useful? \*

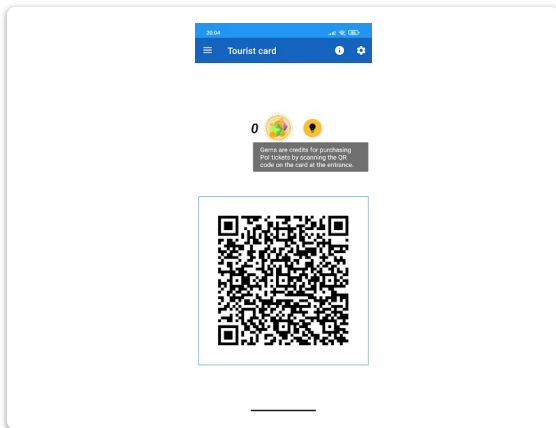
Marcar apenas uma oval.



Scoreboard



Badges



App currency

6. Justify the choice of your preferred gamification element, referring to its usefulness or the sense of encouragement or competitiveness it awakens in you. \*

---

---

---

---

---

- 7. In your opinion, what are the biggest advantages that these gamification elements contribute to the app or its users (aspects related to engagement, reward, motivation for further use)? \*

---

---

---

---

---

- 8. Do you think the gamification elements are good the way they were developed, or would you make any significant changes? If yes, please specify. \*

---

---

---

---

---

- 9. *Remember that you earn +10 XP and +5 gems for every endorsement you issue to your peers and that by activating the peers strategy you will have a better chance of certifying your visit successfully. Also, your privacy is not compromised to your peers and the additional battery that the peers strategy uses is not significant.* \*

Would you activate the strategy that leverages on-site peers if you were to use the app? Justify, referring to whether the gamification elements (or the reward attribution method) had an impact on your choice.

---

---

---

---

---



10. Would you use the app again to visit points of interest? Justify, stating if so which aspect of the app most encourages you to do so. \*

---

---

---

---

---

---

Este conteúdo não foi criado nem aprovado pela Google.

Google Formulários

