TÉCNICO LISBOA

A Blockchain-based Platform for Sharing and Verifying Education Certificates

Diogo Tavares Serranito

Thesis to obtain the Master of Science Degree in

Computer Science and Engineering

Supervisor(s): Prof. André Ferreira Ferrão Couto e Vasconcelos
Prof. Miguel Nuno Dias Alves Pupo Correia

Examination Committee

Chairperson: Prof. José Carlos Martins Delgado
Supervisor: Prof. Miguel Nuno Dias Alves Pupo Correia
Member of the Committee: Prof. João Fernando Peixoto Ferreira

October 2020

Dedicated to my family and friends...

Acknowledgments

Firstly, i would like to thank my thesis supervisors professor Andre Ferreira Ferrao Couto e Vasconcelos and professor Miguel Nuno Dias Alves Pupo Correia, for supporting me throughout all the development of this thesis, and to whom i dedicate the success of this project. Secondly, a recognition to Instituto Superior Tecnico for the demanding but incredibly enriching years, which gave me the necessary knowledge to jump start my career. I am also grateful to the Administrative Modernization Agency for allowing me to apply this project to a real life scenario. Finally, a special thank you to all my family for the incessant encouragement it was given to me during this journey.

Resumo

Diplomas academicos e quali cacoes contrafeitas sao um problema comum em processos de recrutamento. Para alem dos empregadores enfrentarem o desa o de escolher entre todas as candidaturas recebidas, ainda precisam de veri car se os certi cados apresentados pelos candidatos sao autênticos. A tecnologia Blockchain pode oferecer solucoes para este problema, garantindo: e ciência, os dados armazenados na blockchain podem ser recuperados em segundos ou minutos; imutabilidade, ninguem pode adulterar os dados registados; transparência, qualquer um pode validar esses dados; e descentralizacao, nao e necessario con ar numa entidade unica. Esta investigacao faz parte do projeto QualiChain, que visa transformar os processos de recrutamento em organizacoes publicas e privadas, alterando a forma como as quali cacoes sao arquivadas, geridas, compartilhadas e veri cadas. A solucao desenvolvida integra a tecnologia blockchain Ethereum com uma plataforma de ensino superior, permitindo que as organizacoes veri quem a autenticidade e a integridade dos certi cados dos seus candidatos. A *Higher Education App* permitira que instituicoes de ensino superior armazenem na blockchain os certi cados dos seus alunos graduados. As entidades de recrutamento poderao de seguida veri car as quali cacoes dos seus candidatos automaticamente e num per odo de segundos, interagindo com a *Recruiting App*. As instituicoes de ensino superior estao agrupadas num consorcio, podendo votar em todas as decisoes relacionadas com o consorcio atraves da *Consortium App*. Os resultados da avaliacao realizada suportam a expectativa que a adocao desta solucao apresentara substanciais benef cios para qualquer organizacao, nao apenas para impedir certi cados de educacao contrafeitos, mas tambem para aumentar a e ciência nos processos de recrutamento.

Palavras-chave: Certi cados de educacao, veri cacao de documentos, Blockchain, autenticidade, integridade, contratos inteligentes, Ethereum

Abstract

Counterfeit academic diplomas and qualifications are a common problem in recruitment. Not only do employers face the challenge of sorting through all candidate applications, they also need to check if the certificates that candidates present are authentic. Blockchain technology can offer solutions for this problem by assuring: efficiency, data stored in the blockchain can be retrieved in seconds to minutes; immutability, no one can tamper with the data registered on the blockchain; transparency, everyone can validate that data; and decentralization, no trust on a single entity. This research is part of a larger project (QualiChain) that aims to transform the recruitment process on public and private organizations by disrupting the way certificates are archived, managed, shared and verified. The developed solution integrates Ethereum blockchain technology with a higher education platform, enabling recruiting organizations to efficiently verify the authenticity and integrity of education certificates. The developed *Higher Education App* will give the possibility for higher education institutions to store education certificates from their graduated students on the blockchain. Recruiting entities can then verify the qualifications of their candidates automatically and withing a period of seconds by interacting with the *Recruiting App*. Higher education institutions are grouped in a consortium, being able to vote on all consortium related decisions through the developed *Consortium App*. The results of the performed evaluation support the expectation that the adoption of this solution will present substantial benefits for any organization, not only to prevent counterfeit education certificates, but also to increase efficiency in the recruitment processes.

Keywords: Education certificates, document verification, Blockchain, authenticity, integrity, smart contracts, Ethereum

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Digital transformation is the process of integrating computer technologies into various areas of business, and has gained a tremendous importance for organizations to increase e ciency in all their activities. However, academic diplomas and quali cations { *certi cates* for short { are resisting this transformation, as they are still provided in paper or digitized using a scanner.

The recruitment of personnel for an organization can be a lengthy process as it may require analyzing a large number of candidate curricula. Moreover, *verifying* paper or digitized certi cates can be a very time consuming process, as it may involve manually contacting the academic institution that granted them, which is costly in terms of time and resources consumed. However, this *veri cation is necessary*, as there is a market of counterfeit academic diplomas and certi cates. According to a recent survey, 58% of the employers have been caught lying on their curricula and there has been a 33% increase in curricula embellishment and fabrication [Car]. Similar ndings can be found in another recent survey, according to which over half of the curricula and job applications (53%) contain falsi cations and over three quarters (78%) are misleading [Sta]. Even politicians and other public gures are often found at the centre of controversies over their education and certi cates.

A globalized world requires a solution for this problem that goes beyond ad-hoc agreements or national-level structures. Moreover, the solution should be *decentralized*, in the sense that it should not rely on third parties to support it, as no third party can be well-accepted in every country. It also needs to assure certain *cybersecurity* attributes, speci cally authenticity, integrity and availability of the relevant quali cations data. It has to be modular to support expansions, and scalable to support an increasing number of players.

Blockchain technology o ers solutions to these challenges by providing decentralized, immutable (append-only), transparent, and trustworthy storing of data [Nak08, Und16, Pec17]. Furthermore, it allows organizations to remove intermediaries, reducing the time of going through

third-parties, while also lowering the cost of transactions.

In this document, a solution is presented for organizations (e.g., recruiting companies) to *verify the authenticity and integrity of certi cates*. The authenticity property guarantees that the quali cation has been issued by the corresponding *higher education institution* (HEI), whereas the integrity property assures that the quali cation document has not been modi ed after it was issued. To satisfy all the requirements, more than a system we need an ecosystem, that is exible and can grow with the demand. Therefore, our solution is indeed an ecosystem based on a permissionless, open, blockchain. HEIs insert the cryptographic summaries (collision-free hashes [Tsu92]) of the quali cation les in the blockchain, and the full quali cation les in a decentralized le system. These operations are automated by supporting seamless integration with any Academic Management System (AMS) such as Moodle, Blackboard or FenixEdu. With this integration, an AMS can e ciently register a quali cation on the blockchain in a way that makes it easily veri able by any interested organization. We also support revoking certi cates, in the rare cases that may be needed (e.g., it was emitted with an error). Our solution aims to support *a Blockchain-based decentralized ecosystem of universities and other HEIs* at least at European scale, supporting certi cate sharing and veri cation.

The blockchain technology targeted is Ethereum, an open-source, public, blockchain-based distributed computing platform. Since it is a permissionless blockchain, it will allow any interested organization to easily verify their candidate's education credentials. Moreover, this platform also supports *smart contract* (scripting) functionality, crucial for all the system's operations. As mentioned, the quali cation les themselves are stored in a decentralized le system, the *InterPlanetary File System* (IPFS), one of the better known [Ben14]. IPFS uses content-addressing to uniquely identify each le in a global namespace connecting all computing devices through a peer-to-peer network. Similarly to Ethereum, IPFS is decentralized, i.e., there is no central control.

At the current state of the art, and of the practice, the major software engineering challenge of such an ecosystem is arguably to ensure *decentralization*. Despite *decentralization* being one of the main selling points of blockchain and Distributed Ledger Technology, in practice many solutions end up being centralized, or as good as if they were centralized.

This ecosystem is being developed in the context of project QualiChain, a project funded by the European Commission targeting the creation, piloting and evaluation of a distributed platform for storing, sharing and verifying academic and employment quali cations will focus on the assessment of the potential impact of transformative disruptive technologies on the domain of education [Qua18]. The proposed solution solves a Recruitment in Public Sector Pilot

use case from the QualiChain project. This solution makes use of blockchain technology and provides the ability to integrate it with the FenixEdu platform, a modular software platform for academic and administrative management of higher education institutions developed at Instituto Superior Tecnico (IST). With this integration, the FenixEdu platform would be able to e ciently register an education certi cate on the blockchain, that will be easily veri able by an interested organization.

The proposed solution also had the involvement of the Administrative Modernization Agency (AMA), the public institute that exercises the powers of the Prime Minister's O ce in the elds of administrative modernization, simpli cation and digital administration. If AMA or other Portuguese public administration entity wishes to recruit any of IST's graduated students, it can utilize the proposed blockchain to verify their respective education certi cates.

This solution system was implemented and made available on the *GitHub* repository owned by the QualiChain project.[1] It is also being integrated into the QualiChain platform, serving as a strong baseline for upcoming new functionalities. Some of the developed system components were already demonstrated in the rst project review on June 2020.

The system was evaluated through several performance metrics. The results show that organizations can check the validity of a certi cate automatically and within a period of seconds to minutes. These results suggest that its adoption will be highly bene cial for the public institute, since it will result in time and cost savings regarding the validation of candidate applications. Moreover, the HEI will also bene t by improving the quality of service provided to the students, through faster production of certi cates and extension of the services provided by the AMS.

Section 2 starts by giving a brief background on fundamental topics for this research, followed by other use cases for blockchain technology applied to education certi cation in Section 3. The developed solution system and its main operations are then detailed in Section 4. Afterwards, Section 5 de nes how this system was evaluated, by describing di erent indicators that were benchmarked. Finally, Section 6 expresses the conclusions of this research.

Part of this work was published as: \Blockchain Ecosystem for Veri able Quali cations", Diogo Serranito, Andre Vasconcelos, Sergio Guerreiro, Miguel Correia, in Proceedings of the 2nd Conference on Blockchain Research Applications for Innovative Networks and Services - BRAINS 2020, September 28 { 30, 2020, Paris, France.

[1] https://github.com/QualiChain/consortium

Chapter 2

Background

This section starts by presenting the blockchain topic. Two distinct technologies will be compared, permissioned and permissionless blockchains. This subsection is concluded with several blockchain use cases for various areas of society. Afterwards, an open software platform based on blockchain technology, Ethereum, will be introduced and explained. This document dedicates an entire subsection to Ethereum, since it was the blockchain technology adopted in the proposed solution. After that, the Fenix Academic Information Platform and its functionalities will be discussed, presenting separate subsystems that allow this platform to manage security, authentication and back-office operations. Finally, the structure of an education certificate is presented, along with different ways for recruiters to verify its authenticity.

2.1 Blockchain

Blockchain technology is based on a distributed ledger structure composed by cryptographically chained blocks. This ledger is independent from any central authority and is shared between the distributed computers on a peer-to-peer network [Und16]. Each new block is formed by a set of records/transactions that will be attached to the end of the chain.

2.1.1 Cryptocurrencies and Smart Contracts

Cryptocurrencies were the first application of blockchain technology, that used a public ledger for monetary transactions with minor programmable capability [XPZ+16]. Bitcoin was the first practical solution, where the blockchain would be replicated and accessible on all the network.

Whenever a user requests a new transaction, it needs to be verified by other computers (miners) in the network. If this request is validated, miners will aggregate it to a new block. Each block contains a hash pointer which links to the previous block. This way is impossible

to insert, delete or modify any block in the middle of the chain without recomputing all hashes from the following blocks [ZS18]. If someone were to perform any modi cation, the hash stored in the next block would not match with the modi ed block hash.

Miners are constantly receiving noti cations of transaction requests and collecting them to create a new block. This leads to competition between each other, because the rst to generate a valid block gets paid for his service [Pec17]. However, there is still the possibility for a miner that wants to insert a modi ed block in the middle of the chain, to recalculate the hashes of all the following blocks. To solve this problem, a block can only be agreed upon and attached if a hash function of the block (proof-of-work) has been resolved satisfying mathematical conditions [Vra17]. Therefore, recalculating all the following blocks would take much more time and computational power. Thus, immutability is one of the main principles of blockchain technology.

There are several di erent blockchain consensus algorithms, each one having its own advantages and disadvantages. Currently, *proof-of-work* is the adopted algorithm in Bitcoin and Ethereum [Nak08, Eth14]. This algorithm creates a computational power competition, in which participating nodes have to solve a cryptographic puzzle to win the right to create a new block, and be rewarded accordingly [ZL19]. *Proof-of-stake* is another popular consensus algorithm. Rather than depending on computational power, this algorithm selects the node that will create a new block by its respective stake [ZL19]. It is also an energy-saving protocol, since it encourages the acquisition of the internal currency instead of consuming a large amount of computational power to reach a consensus [ZL19]. Alternative consensus protocols would be *Practical Byzantine Fault Tolerance*, *Delegated Proof of Stake*, *Ripple*, among others [ZL19].

After the Bitcoin debut, it was clear that such a system would be more useful than just money transactions. People started to envision what other kind of applications could run on a blockchain. If miners were already running small programs to validate transactions, why not run more complex programs [Pec17]?

The second generation of blockchain was developed around a programmable infrastructure, starting with Ethereum. Simple currency transactions were replaced by more versatile autonomous programs running across the blockchain network, named Smart Contracts. Users can interact with these programs by sending transactions with speci c instructions to be processed.

Ethereum replaced simple currency transactions for more versatile autonomous programs running across the blockchain network, named smart contracts. Users can interact with these programs by sending transactions with speci c instructions to be processed. *Smart contracts* are self-executing scripts that reside on the blockchain that allow for distributed and heavily

automated work ows [CD16]. These *contracts* are deployed and executed on the blockchain network and can be used to reach agreements and solve common problems with minimal trust. It is possible to build self-executing *contracts* on the Bitcoin blockchain network, however, these *contracts* are very simple and limited due to the corresponding scripting language, which does not allow for complex control ows [XPZ+16].

2.1.2 Permissioned and Permissionless Blockchain

Blockchains can be either permissioned, with a private restricted group of participants, or permissionless, allowing public use [Und16].

Permissionless blockchains have no owner, and everyone can contribute and retain a copy of the ledger. The integrity of the ledger is guaranteed by its participants through reaching a consensus between them [UK 16].

Any user can freely begin interacting with the network, since there are no restrictions on which users can submit transactions or create new blocks. Furthermore, everyone has the possibility of running a node and operating mining protocols for transaction veri cation.

The biggest advantages related with permissionless blockchains are:

Permissionless networks need to be decentralized, meaning that no central authority is able to change or rewrite any part of the ledger.

Anonymity is another important attribute, not requiring users to submit personal information when creating an address or performing a transaction. However, due to legal reasons, personal information is occasionally required.

Transparency is a requirement for users to trust the network. Therefore, there is the need to give users access to all information in the blockchain without restrictions.

Bitcoin and Ethereum are the most popular examples of permissionless blockchain systems, where miners (veri ers) are crucial to prevent participants from inserting or modifying the ledger with transactions that are against the rules. For a transaction to enter the ledger it needs to be veri ed beforehand.

With the growing popularity of permissionless blockchain technology, nancial institutions began to realize that the open structure was against their needs. Users being only represented by an alphanumeric public address with no evidence of their real identity was a major concern for these institutions, since several banking laws prohibit anonymity for nancial operations [Pec17].

Permissioned blockchains were created to solve these concerns, by developing a blockchain in which transaction processing is carried out by a well-de ned set of identi ed subjects, and all data in the system may only be visible for a limited group of entities.

This kind of system is intended for entities that already have at least a small degree of trust between each other, therefore, only a prede ned set of subjects can add new blocks to the ledger, meaning that there is no need for proof-of-work mining or transactional tokens (cryptocurrencies) anymore [Pec17].

By adopting a permissioned blockchain system, nancial institutions could grant limited read access to transactions and block headers to their clients, providing a transparent and trustworthy way of assuring the security of their client's funds. Full read access can also be given to regulators in order to meet a required level of compliance. This would facilitate independent auditing by regulatory entities [Gar15].

Permissioned blockchain technology goes beyond nancial applications, with other industries that want to protect important customer data embracing similar systems.

Many projects regarding permissioned blockchains are built based on *Hyperledger Fabric*, a blockchain framework implementation of a distributed ledger platform for developing applications with a modular architecture, hosting and running smart contracts, being one of multiple projects currently in development under the *Hyperledger* Project [Cac16].

2.1.3 Blockchain Security Risks

Blockchain is currently considered a breakthrough technology, that could be applied to a wide variety of domains in everyday social and business activities. However, even with blockchain technology representing innovation and a large set of opportunities, there will still be risks associated with its use.

Control over a decentralized network is one of the main concerns regarding the use of blockchain. Sybil attack is one possible threat to this control, where there is an attempt to control a peer-to-peer network by creating multiple fake identities [ZS18]. These fake users appear to be unique. However, a single entity controls many identities at once and, consequently, can in uence the network through additional voting power. *Proof-of-work* helps mitigate this risk, because without having computational power, nodes cannot add blocks to the blockchain data structure.

Another threat to the lack of control over a decentralized network would be the 51% attack. This attack consists of a group of nodes having 51% of the computational power in the network. As a result, these nodes would have the possibility to con rm transactions and create new blocks.

This attack is not economically feasible, since there would be the need for a vast computational power to implement it [ZS18].

Furthermore, *smart contracts* also possess dangerous vulnerabilities which malicious users or miners can exploit to gain pro t. *Reentrancy* in smart contract functions can be a problem, since this type of function can be interrupted during its execution and reinvoked without its previous call being completely executed [KGDS18].

Another vulnerability present in smart contracts is the *Transaction State Dependence*. In a chain of contract calls, a contract should only be able to know the address of the contract from which it was called. However, the *tx.origin* state variable can be used by a contract to obtain the initial contract that started the chain [KGDS18], which results in a privacy related problem.

If a contract has a *Block State Dependence* in its implementation, it depends on variables that are de ned in some block's header. Therefore, a block miner is able to tamper with a block by modifying certain values to favor himself on the vulnerable contract [KGDS18]. This attack has various degrees of success since a miner can not guarantee its own selection to be the miner of a speci c block [KGDS18].

Moreover, a contract can also have a *Transaction Order Dependence* if its functionality depends on a correct processing of concurrency. Consequently, a miner can in uence the order of transactions in a new block, resulting in unexpected results in a contract's state that might be nancially favourable to the miner.

2.1.4 Blockchain Use Cases

Due to Bitcoin's popularity, blockchain started to gain recognition, and di erent proposals to the application of this technology started to emerge. Since this technology is in its early adoption stage, a large amount of unrealistic and impractical solutions are still often proposed. There is the need to clarify and clearly understand all of blockchain's possible bene ts, limitations and real-life applications [ZS18].

Financial applications make for the largest number of attempts in the use of blockchain technology. With the emergence of Bitcoin and other cryptocurrencies, the nancial sector quickly realized the opportunities associated with the adoption of this kind of technology, by taking advantage of blockchain's transparency, enhanced security, increased e ciency, improved traceability, immutability and reduced costs. Even though nancial applications are showing great potential, there are still a large number of issues that need to be addressed, for example, data privacy and scalability in nancial markets.

Supply chain management is one of the areas that blockchain could possibly be applied with great success. Transparency is a huge problem companies come up against when managing their supply chain. There is an enormous difficulty for customers to truly know the value of products because there is a significant lack of transparency in the supply chain network. It is also extremely complex to investigate supply chains for suspicion of illegal or unfair practices. Considering the immutability, reliability and integrity guarantees given by blockchain technology, its use could increase significantly customers confidence and satisfaction when purchasing a product.

The decentralization of the energy market is another area that blockchain can possibly be applied. Currently, large corporations are controlling all the energy distribution in each market. Blockchain technology has the potential to provide innovation in peer-to-peer energy trading and decentralized energy generation [ARF+19]. An individual's energy produced through a solar panel can be quantified, traded and recorded in a ledger, allowing him to be both a consumer and a producer of energy, creating a decentralized market where prices would not be defined by large corporations. This could significantly improve efficiency and reduce costs.

Healthcare is a field where blockchain can completely transform the way critical data is stored and protected. In today's society there is a growing demand for instantaneous access to important healthcare information. However, patient data is stored inconsistently across multiple medical institutions, resulting in a difficult access to a patient's standardized data. With the use of blockchain technology, a consistently up to date decentralized database regulating medical data access could be adopted. In each medical process, there are a large variety of different parties involved. This leads to a time-consuming identification and authentication of the varied medical stakeholders. Blockchain technology can offer more efficient and accelerated stakeholder registration in every medical procedure, resulting in a better and more effective healthcare system [Met16]. Blockchain also grants the security of patient's health information through data encryption, allowing to set a conditional access control system where only authorized stakeholders can view patient's medical records. Every access to this data will be registered in the ledger, giving patients more control over their own heath information.

2.2 Ethereum

Bitcoin has been acclaimed as a revolutionary development in money and currency, being the first example of a digital resource which has no intrinsic value and no centralized controller.

However, a big interest for the underlying technology (blockchain) quickly emerged.

There were three main approaches to building advanced and complex applications on top of a cryptocurrency. The first was building a new blockchain system, that would allow unlimited freedom in building a feature set, at the cost of development time, effort and security. The second was using scripting on top of the already established Bitcoin blockchain, that was easy to implement but very limited in its capabilities and scalability. The third and last approach was building a meta-protocol on top of Bitcoin technology, that while also easy to implement, suffered from faults in scalability [Eth14].

Ethereum was created with the intent to be a solution for this problem, offering an alternative protocol for building decentralized applications, with fast and accessible development, efficient interactions between different applications and even security guarantees for small-scale applications [Eth14].

Ethereum uses the blockchain technology to provide a built-in fully fledged Turing-complete programming language that can be used to create *contracts* that can be used by any user to create any system, simply by writing up the logic in a few lines of code [Eth14].

While in Bitcoin transactions identify monetary payments, Ethereum transactions can store monetary value, code, and/or parameters and function calls [XPZ+16]. This transaction will be signed with the sender's private key, assuring authentication and authorization for transferring money, creating a contract, or pass the data parameters associated with the transactions. This transaction needs to be valid and properly formed to contain all the information needed to be executed.

The Ethereum blockchain state is composed of objects called *accounts*. State transitions are direct transfers of value and information between these *accounts*. Each *account* contains a nonce that acts as a counter to assure cryptographic freshness, being used to make sure each transaction can only be processed once. It also contains the account's current *Ether* balance, contract code (if present) and storage [Eth14].

Ether is the underlying token powering the Ethereum blockchain, that acts as the fuel of Ethereum and is used to pay transaction fees.

There are two types of *accounts* in Ethereum:

Externally owned accounts, controlled by private keys and possesses no code. Anyone can send a message from this type of account by creating and signing a transaction using his private key.

Contract accounts, controlled by their respective contract code, so if the *account* receives a message, this code activates, allowing for read and write operations to internal storage.

It also authorizes the *account* to send other messages or create contracts.

Externally owned accounts can commit transactions (di erent from Bitcoin transactions) that contain the following structure:

From: 20-byte address corresponding to the account sending the transaction.

To: 20-byte address identifying the recipient of the transaction. It can represent a *Externally Owned Account*, a *Contract Account*, or just the null address ("0x00") to deploy a new contract.

Value: the amount of tokens in *wei* to be transferred (10^{18} *wei* = 1 *Ether*).

Data: eld containing data necessary for contract execution or deployment.

STARTGAS: the limit number of computational steps the transaction execution is allowed to perform.

GASPRICE: the fee the sender is willing to pay per computational step.

The last two elds are crucial to prevent accidental or hostile denial of service attacks by setting a limit of computational steps the contract can use, avoiding in nite loops or excessive computational wastage [Eth14].

Ethereum uses a fee system operating with the computational unit *gas*. If an attacker were to perform a denial of service attack, he would have to pay proportionately for each resource consumed, leading to a massive increment in the *gas* fee.

Ethereum's messages are similar to transactions, except they are produced by a *contract account* instead of an *externally owned account*. These messages are virtual objects that only exist in the Ethereum execution environment and allow contracts to communicate and maintain a relationship between each other.

Ethereum implements an execution environment on the blockchain called the Ethereum Virtual Machine, running on the nodes participating in the network. Ethereum virtual machine code, a stack-based bytecode language, is written in each contract, where each byte represents an operation to be executed [Eth14].

To verify a block, every node in the network goes through the transactions listed in the new block and runs the code that was triggered by each transaction within the EVM. Each node will then perform the same calculations and store the same values. This allows the network to reach a consensus on the system state, in a more e cient way.

2.3 Fenix Academic Information Platform

FenixEdu is a modular software platform for academic and administrative management of higher education institutions.

The FenixEdu project started in 2002 at Instituto Superior Tecnico. IST started developing information systems to manage its internal information back in the 80's, and around the year 2000 it became clear that those systems could not grow as fast as the school wanted them too. The internet was developing and expanding, and the old systems were written in outdated platforms without worldwide network access. Creating a business strategy that took into account their previous experience with building IST's information systems, a new system, Fenix, was designed and has been in continuous improvement since then [S+11].

The Fenix system was developed to be an integrated platform that could be applied to all levels of the academic process. It implements three major management systems:

Content Management System (CMS) that can be used to administer course, degree, department and institution content.

Student Management System (SMS) for educational institutions to manage student data. This data includes attendance, behavioural information and assessment information.

Learning Management System (LMS) used for administration, documentation, tracking, reporting and delivery of educational courses and development programs [Ell09].

Fenix o ers management and support for all academic tasks, such as online student applications and enrollment; evaluation, grades and student curriculum archive; full management of academic records; teacher and course scheduling; room reservation and availability; and many other functionalities [S+11].

The Fenix system also provides support to academic back-o ce operations, including management of process work ows. Functionalities as degree and course planning; European Credit Transfer and Accumulation System (ECTS) validation; issue of diplomas and certi cates; publication and selection of these theme proposals and their respective supervision, are some of the capabilities that this system provides.

Fenix academic platform integrates a identity management subsystem for user authentication to all computer and network services. This subsystem is powered by Kerberos, a network authentication protocol designed to provide strong authentication for client and server applications through secret-key cryptography, and Lightweight Directory Access Protocol, a client/server protocol used to access and manage directory information [S+11]. This identity management subsystem also supports authentication based on the Portuguese Citizen Card, allowing digital

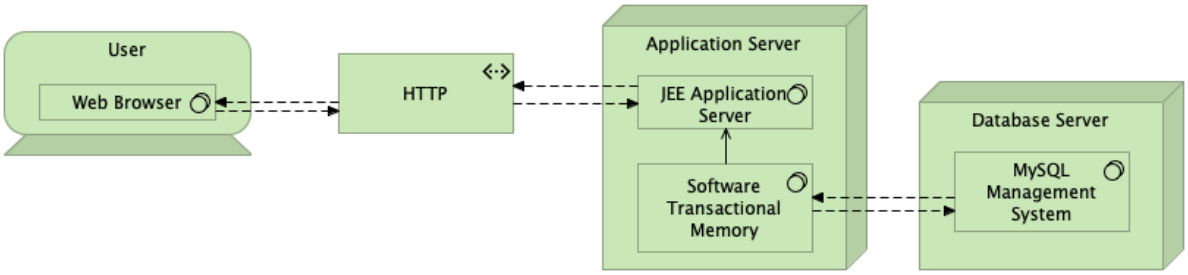

Figure 2.1: Fenix Software Architecture

signing of operations and documents. Furthermore, it integrates with the European electronic-ID interoperability subsystem.

The original software architecture of Fenix consisted of a regular three-tiered architecture: web browser, application server and relational database server. A user interacts with the web browser, which communicates with the application server through HTTP requests. When processing a request, the application server, implemented using the Java platform, usually requests or stores data in the database server, supported by the MySQL management system [S+11].

Fenix domain model is considered rich, because it is composed by many di erent entity types and slight variations between them. With the development of such a complex model, some of the services were taking too long due to excessive read/writes in the database. Therefore, a new software architecture was developed to address this issue. The key element was to implement a Software Transactional Memory (STM) in the application server, as illustrated in *Figure 1*, that only relied on the database to assure durability from the ACID properties [S+11].

The development of the Fenix Framework was another of the key aspects in the new Fenix architecture that contributed to its success. The domain model was de ned using a new language, the Domain Modeling Language (DML), that was speci cally designed to implement the structure of a domain model [S+11].

Fenix Framework is a result of research work that was done in the context of the Fenix project. It is used as a tool for students to develop new applications in the IST's degree on Information Systems and Computer Engineering. It can also serve as a basis for further research done by Masters and PhD students, as well as by senior researchers on several research projects.

When a student graduates, he requests the emission of his diploma to the graduate area service. A university's sta member interacts with the Fenix system to generate the diploma in PDF format. Then, this document is printed on paper, signed and stamped. However, Fenix still does not o er the possibility for students to digitally receive their diplomas, although the registration certi cates are already digitally signed and distributed electronically.

2.4 Education Certificates

An education certificate is a document that certifies that a person has received specific education and/or evidence of achievement of expected learning outcomes. Education certificates are used for a variety of purposes, as the recognition of the completion of a specific learning experience by a student; the achievement of a defined amount of learning achieved in a specified area; the acquisition of skills or attaining a particular excellence criteria.

Every certification involves the following elements [GC17]:

Claim: statement of what fact is being certified. For example, \a student has completed an assignment and achieved the respective expected learning outcome".

Issuer: identification of who is validating, confirming and certifying the facts.

Evidence: proof supporting the claim, indicating the procedure by which the claim was verified, and other important information endorsing the legitimacy of the certification.

Recipient: the individual addressed by the claim.

Certificate: a document affirming the identity of the issuer and recipient, the claim and the evidence.

Signature: a unique symbol/code that only its respective issuer can append, assuring its identity.

After the education certificate has been issued, the recipient can now share it with a third-party, such as an employer. Subsequently, this certificate needs to be verified to assure its authenticity.

This can be accomplished with three different methods [GC17]:

Verifying specific ingrained security features on the certificate, such as checking the correctness of the signature/seal.

Contacting the original issuer, inquiring if the certificate was indeed issued by them.

Comparing the certificate with a centralized database, where the issuer stores the copies of its issued certificates.

There are many diverse proposed solutions on this topic, using different platforms with blockchain as the underlying technology. These solutions will be presented in Section 3.

2.5 Summary

This section presented and explained the core technology of the developed system, the blockchain. Several use cases for various areas of society were also described. Afterwards, an open software platform based on blockchain technology, Ethereum, will be introduced and explained. Ethereum, the blockchain technology adopted in the proposed solution, was then introduced and explained. After that, the Fenix Academic Information Platform and its functionalities were discussed. Finally, the structure of education certi cates and their respective authentication processes were presented.

Chapter 3

Related Work

In this section we will explore the application of blockchain technology to education certi cates. Three distinct projects that explore di erent facets of this topic will be presented, nishing this section with a discussion on important aws identi ed in each one.

3.1 Proof of University Diplomas in Greece

Cardano is a smart contract platform, similar to Ethereum, with the intent of running nancial applications. This platform was built through a layered architecture to o er scalability, interoperability and sustainability.

Cardano's rst use case will be to prove the veracity of university diplomas in Greece. GRNET, the national research and education network of Greece, in a partnership with IOHK company, are responsible for developing this project [Cas18]. This application will be developed on *Enterprise Cardano*, a permissioned ledger version of *Cardano*, allowing only a private restricted group of participants.

Considering the labor-intensive process of con rming a certi cate, this project aims to simplify it by inserting these student diplomas on a blockchain. This would allow for a easier and faster method for checking if someone really holds a degree. GRNET intends to only insert cryptographic hashes of the certi cates, with the purpose of protecting student data privacy. Even if unnoticeable subtle changes were made to the diploma, an employer could check the authenticity of the document by comparing its hash with the respective one stored in the blockchain.

GRNET states that one the main di erences from similar projects is the fact that it stores each veri cation step in the blockchain as a transaction [Cas18].

3.2 Blockcerts

Another effort to store diplomas on the blockchain comes from a partnership between MIT Registrar's Office and Learning Machine, a software development company [DT17]. Massachusetts Institute of Technology graduates can now receive their diploma through an app. This is app is called *Blockcerts Wallet*, and allows any student to obtain a tamper-proof diploma, that can be verified by any employer or school.

In 2016, the *Blockcerts* toolkit was developed as an open-source toolkit created using Bitcoin's blockchain technology, aiming to grant the possibility of universities being able to issue their graduates education certificates. In order to secure each graduate's education credentials, there was the need for each student to generate and store their respective public and private keys. This approach would be unfeasible and unpractical, considering the lack of knowledge of students about blockchain and cryptographic keys, creating a huge barrier for its effective use.

The app *Blockcerts Wallet* was created to address this issue. When a student downloads and installs the app, it generates his own keypair, sending the respective public key to MIT, that stores it in a digital record. A hash is generated from the student's diploma and added to the blockchain. Finally, MIT sends to the student his certificate, with his public key registered into it, allowing him to easily prove the diploma's ownership with its private key. Furthermore, an employer can efficiently verify if a digital diploma is legitimate, through a portal where the diploma can be uploaded, and its hash compared with the one stored in the blockchain.

This project is still in development and there is still space for modifications. The loss of a student's private key through deleting the app is a serious problem, since the student is not able to prove the ownership of his educational certificate anymore.

3.3 CredenceLedger

CredenceLedger is a proposition for a permissioned blockchain-based platform for decentralized verification of academic credentials [AF18]. This project is based on Multichain, an open platform for building blockchains, that helps organizations to quickly build and deploy applications [Gre15].

Using *CredenceLedger* mobile app, students can access their educational credentials, that can be easily verified by employers. Privacy is one the most important features of this solution, therefore, digital certificates are protected in order to only be available limited information about them. If a third party is interested in obtaining more information about a certificate, it can perform a request that will be subject to the owner's approval [Gre15].

Adopting a permissioned blockchain, allows for this project to maintain a few important features from permissionless blockchains, but adds regulated system functionalities, with a controlled set of rules and permissions.

3.4 Discussion

After analyzing several different proposals for the application of blockchain technology to education certificates, it is clear that no description of a similar solution is available today.

There are a few preliminary proposals of systems for storing certificate data in a blockchain, e.g. the solutions described in Sections 2.1 and 2.3, but they are focused on a limited environment, not in providing a decentralized, scalable ecosystem.

A noteworthy case is MIT's Blockcerts toolkit and wallet (Section 2.2), that allows their graduates to receive their diplomas in a digital form [DT17]. However, even if this platform provides the possibility to verify that a certificate is authentic, it does not assure that the certificate's issuer is trustworthy. Furthermore, for verifying a certificate's authenticity it is required to check if the certificate is included in the awarding institution's certificate revocation list. This process has to be repeated for each certificate that needs to be verified.

Chapter 4

Blockchain Ecosystem for Veri able Quali cations

This section will start by presenting an overview on the QualiChain project architecture. Then, it will introduce the Recruitment in Public Sector Pilot use case, while also presenting an assessment on AMA's recruitment process. Finally, it will propose a reference solution applied for this use case, based on Ethereum blockchain technology integrated with the FenixEdu platform.

4.1 QualiChain Architecture

The QualiChain project started in January 2019, focusing in the design, implementation, piloting and thorough evaluation in terms of bene ts, risks and other potential implications of disruptive technological solutions. This solution aims to be a distributed platform targeting the storage, sharing and veri cation of academic and employment quali cations. This project investigates the transformative impact of disruptive technologies, such as blockchain, semantics, data analytics and gami cation in the domain of education certi cation.

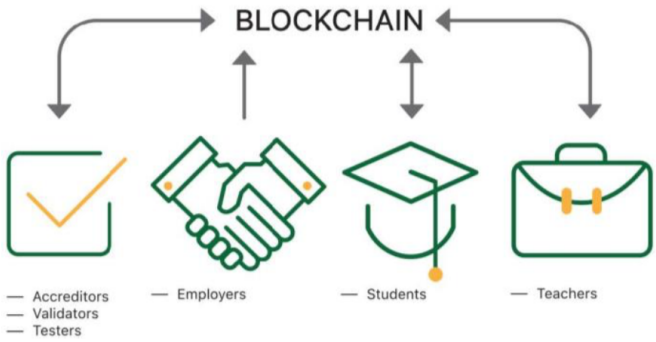

Figure 4.1: The value of blockchain to QualiChain stakeholders [Qua18]

As illustrated in Figure 4.1, QualiChain aims at implementing blockchain technology to create a distributed platform to allow accreditors, validators and testers to issue, validate and revoke academic and employment quali cations. Students and teachers would be able to con rm, manage and share their personal education credentials, which employers can easily verify without the need for a complex process of contacting the issuing organization.

QualiChain services are structured along two main pillars, *Baseline Services* and *Value Adding Services*.

Baseline Services are based upon QualiChain main technological foundations, blockchain and semantics, being described by three main services:

Awards'/ Quali cations' Archiving, enabling public education institutions, as well as other accreditation authorities to store academic and other quali cation awards on the blockchain. This approach aims to revamp and modernize the former organizations' archiving and management practices, resulting in time and cost savings both for organizational entities as well as for their employees.

Awards'/ Quali cations' Veri cation, allowing employers, recruiters and human resources departments to e ortlessly validate claims around the possession of degrees, academic titles and other certi cations, while relieving issuing organizations from the task of dealing with employers' certi cation queries.

Quali cations' Portfolio Management, giving individuals access to an area where they can manage and share their veri ed achievements, quali cations and experience.

Value Adding Services are supported by the *Baseline Services* to o er a set of more advanced services, such as career counselling, intelligent pro ling and competency management.

To implement the services previously described, the QualiChain platform logic layer, represented in Figure 4.2, consists of three main components:

Validation and Veri cation Engine, responsible for registering newly awarded certi cates and achievements as well as for ratifying claims around the possession of certain awards and quali cations. This component features an *Awards' Registration Interface* that allows for organizations to issue and register newly veri ed quali cation records in blockchain's distributed ledger. It also contains a *Validation Query Builder* through which all issuing institutions and individual users can set up appropriate validation queries.

Pro ling and Career Management Engine, featuring a *Portfolio Manager Interface* that makes accessible functionalities required for the management of individual users'

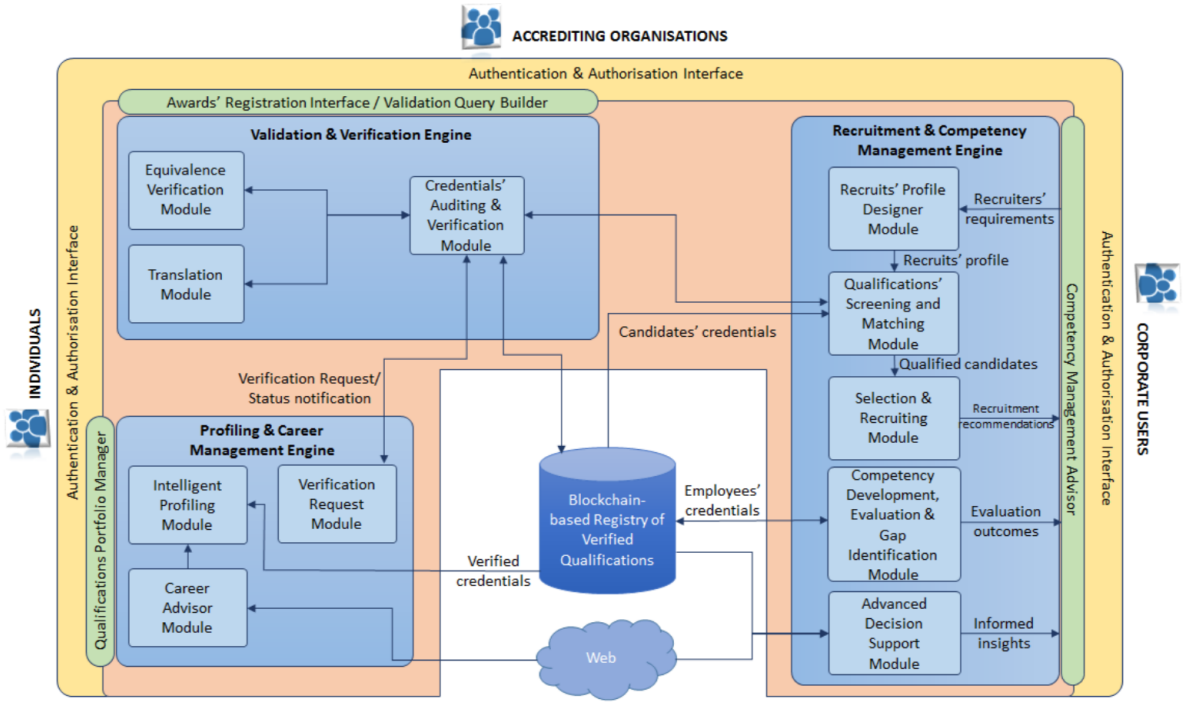

Figure 4.2: QualiChain High-level Architecture [Qua18]

digital portfolio. Users can archive and access their achievements, quali cations and work experience with the purpose of showcasing them to third parties.

Recruitment and Competency Management Engine, o ering functionalities for competency management at both strategic and tactical level addressed to corporate users, to not only education providing institutions, but also public authorities, private companies and policy makers. This component exposes its functionality through a *Competency Management Advisor Interface*.

The QualiChain data access layer intends to store and retrieve quali cations' data from blockchain records kept on a *Blockchain-based Registry of Veri ed Quali cations*. Considering that education certi cates usually occupy a large amount of data, storing the complete certi cates on the blockchain would be an ine cient approach. Therefore, there is still the need to store the complete certi cates on the platform. This can be accomplished by adding a storage component to the architecture (not represented in Figure 4.2), where all education certi cates would be archived and managed.

4.2 Recruitment in Public Sector Pilot

One of the use cases that will be analyzed by the QualiChain project is using the its platform and services to support and simplify the public sector recruitment and competency management

Figure 4.4: Current Recruitment Process at AMA

procedures. The platform described in this document can be applied to this use case, through a pilot project involving the Administrative Modernization Agency (AMA) and INESC-ID, which is one of the research centres of Instituto Superior Tecnico (IST).

IST graduates engineering students, some of which are contracted by AMA and, through AMA, by several Portuguese public administration entities. Therefore, AMA is interested in having access to certi ed diplomas from those alumni who apply to such positions, as illustrated in Figure 4.3.

Figure 4.3: Use of QualiChain for sta ng the Portuguese public sector

As presented in Section 2.4, FenixEdu is a modular software platform for academic and administrative management of higher education institutions developed at IST, that operates in various higher education schools. This pilot project aims to integrate QualiChain with some of the FenixEdu application programming interfaces, in order to access the education certi cates. The development of this pilot will try to implement blockchain capabilities to the FenixEdu platform, demonstrating a real use of that solution applied to IST student certi cates.

The current recruitment process was assessed in a collaboration with AMA, and is illustrated in Figure 4.4, represented in aBusiness Process Model and Notationdiagram [Gro14]. When-

ever someone decides to apply for a job opening at AMA, they need to prepare and send their respective resumes and education credentials. AMA will then analyze all applications received and create a shortlist with the best possible candidates. The latter are invited to an interview where their technical and \soft" skills are assessed through a series of evaluations. After completing the necessary interviews, AMA determines the candidate(s) that better match the requested profile. If further information about a candidate's education credentials is needed, the latter will be contacted to provide the incomplete data. After having all the mandatory information about the candidate, the latter will receive a job proposal from AMA. However, recruiters blindly believe that the received education certificates are authentic, not performing any further verifications.

4.3 The Blockchain-based Platform

This section presents the design of our Blockchain-based platform for storing, sharing and verifying education certificates. The solution essentially allows HEIs to push their certificates into the platform, and recruiters to check if a certificate was issued by a certain HEI (authenticity) and that it has not been tampered with (integrity).

4.3.1 Solution Overview

Most academic certificates today are still issued in paper, then sometimes digitized to a PDF or similar format. Digital signatures based on public-key cryptography [RSA78] can be used to provide the same or more security than the paper-based solution. However, digital signatures have to be verified with a public key, that must be bound to an identity and distributed in a secure way, i.e., its authenticity and integrity must be guaranteed. This problem is known and is solved with a Public-Key Infrastructure (PKI), that must include a set of organizations designated Certificate Authorities (CAs) [IT00]. CAs issue digital certificates (not to be confused with academic certificates) that contain bindings between a public key and an identity. This is a heavy solution with several inconveniences, one of them the centralized nature of CAs.

With the introduction of blockchain technology, this project aims to make education certificates accessible and easily verifiable by any employer or interested entity, making certificate verification processes faster and more efficient. The use of blockchain also assures decentralization, meaning that HEIs and recruiting organizations do not have to trust centralized organizations like CAs. Furthermore, there is less likely to be a single point of failure in a decentralized system. This project decided to adopt Ethereum, a permissionless blockchain technology. This choice allows for any institution/organization to possess a node connected to the blockchain network.

Figure 4.5: Solution context.

Ethereum supports smart contracts, that we use to register and revoke certi cates, among other operations.

The project's data access layer aims to store and retrieve certi cate data from Blockchain records kept on aBlockchain-based registry of veri ed quali cations. Considering that education certi cates usually occupy a large amount of data, storing the complete certi cates on the blockchain would be an ine cient approach, since, for each update to the distributed ledger, all nodes in the network would also need to store their respective copies of each certi cate. This would be computationally expensive resulting in considerable fees for each registration. Therefore, the blockchain only stores metadata and a cryptographic hash of each certi cate, substantially reducing the size of the ledger. This is possible due to the collision-resistant nature of hash functions, i.e., to the practical impossibility of nding two di erent inputs (e.g., two di erent certi cates) that have the same hash (the output of the function). However, there is still the need to store the complete certi cates on the platform. This is accomplished through IPFS, a peer-to-peer distributed system for storing and accessing les that seeks to connect all computing devices with the same system of les [Ben14]. This decentralized le storage system allows complete education certi cates to be archived and managed.

Figure 4.5 illustrates the solution context of the proposed solution. This gure is represented in ArchiMate Enterprise Architecture modeling language, a visual language for describing, analyzing, and communicating many concerns of Enterprise Architectures [The19]. When a student graduates at a HEI, the latter will process and issue the education certi cate corresponding to the achieved quali cation. Afterwards, this certi cate is registered on the blockchain and submitted to the IPFS network. The graduate student can then share its earned certi cate

with employers in recruitment processes, that will verify the certi cate's legitimacy through the blockchain. Furthermore, the graduate can also share the multi-hash acquired from IPFS when submitting the certi cate to the system. The recruiter can then retrieve the complete credential associated with the received multi-hash and verify its legitimacy.

All entities interact with the Ethereum blockchain through a JavaScript application. Three di erent modules were developed and adapted to the needs of each class of entities:

Higher Education App: this module o ers the possibility for registration and revocation of education certi cates.

Recruiting App: this module allows recruiters to easily verify a candidate's education certi cate.

Consortium App: this module allows new HEIs to join the platform through a consortium-based voting system.

In the following sections, these modules will be discussed and analyzed by presenting multiple approaches that could be taken to implement this solution. Additionally, the usage of smart contracts in this project will also be presented and described.

4.3.2 Consortium Smart Contract

Each consortium includes a set of Higher Education Institutions. For instance, project QualiChain is boot-strapping a consortium with a few HEIs, but eventually more HEIs will join and other independent consortiums may appear. An ecosystem with a single consortium and three HEIs is represented in Figure 4.6. Data related to the consortium is stored on the blockchain in a Consortium smart contract. This smart contract contains information about the HEIs that are part of the consortium and the rules of the consortium (e.g., how a HEI can join).

Without compromising decentralization, this project considered that there is an entity that jumpstarts a consortium by deploying a consortium smart contract on the Ethereum blockchain network. This smart contract was developed in Solidity, an object-oriented, high-level language for implementing smart contracts, the most adopted for Ethereum [Eth19]. Ethereum does not run Solidity contracts, but Ethereum virtual machine (EVM) code obtained by compiling the respective Solidity code.

Contract structure

As represented in Figure 4.7, a consortium smart contract de nes a mapping (a key-value data structure) named contracts that associates a HEI's Decentralized Identi er (DID) to its respec-

Figure 4.6: Data ow diagram representation of an ecosystem with a single consortium of three HEIs.

tive smart contract address. For HEI identi cation, this project adopts DIDs as de ned by the W3C [W3C]. These identi ers are designed to allow for an entity, e.g., a HEI in our case, to prove control over its respective DID, and are implemented independently of a centralized registry, identity provider, or certi cate authority (CA). DIDs are essentially URLs that relate a subject to means for trustable interactions with that subject. For this system it was decided to use DIDs in the form did:ethr:address, where ethr represents a Ethereum based method and address corresponds to the HEI's account address.

Governance decentralization is one the properties assured in this project, since it is up to the consortium to decide if a HEI is allowed to join or leave, similarly to what happens in many organizations, such as private associations. When a HEI wants to join the developed platform, it deploys its own smart contract (see next section) and it makes a request to join the current consortium of HEIs. The latter will decide if a new HEI should join, through a voting system implemented in the Consortium smart contract. The emphasis on decentralization is well de ned through this voting system, since every change to the contract's state needs a successful decentralized voting of independent entities. Therefore, there is no need for a centralized entity to manage and supervise the actions being performed on the platform.

The Consortium smart contract de nes three main operations to interact with the voting system and to obtain the resulting state:

Figure 4.7: Attributes, functions and relations between a Consortium Smart Contract and a Higher Education Smart Contract. Represented in Uni ed Modeling Language [Gro17].

registerHEI( ) : vote to register the contract address of a new HEI joining the platform.

cancelHEI( ): vote to delete a HEI's contract address to cancel its subscription to the platform.

changeThreshold( ) vote to replace the current threshold with a new value.

getHEI( ) : return a HEI's contract address corresponding to the received identi er.

The functions registerHEI() , cancelHEI() and changeThreshold()are restricted to only being called by Ethereum accounts owned by HEIs that are part of the consortium. Therefore, it is assured that only members of the consortium can vote.

Each HEI can vote to add a new member to the consortium, by calling theregisterHEI( ) function with the respective HEI unique identi er as a parameter. If this function was called for the rst time for a distinct HEI identi er, a new element is added to the registerPolls mapping. This mapping links the identi er of a HEI, currently being voted to join the consortium, with a HEIVote structure that de nes ve di erent elds:

contractAddress: address of theHEI smart contract deployed by a new HEI.

positiveVoteCounter: number of votes for the admittance of a new HEI in the consortium.

negativeVoteCounter: number of votes against the admittance of a new HEI in the consortium.

voters: mapping that associates the identi er of HEIs with their respective vote value. A positive vote corresponds to the value 1, the value 2 represents a negative vote and if a HEI has not voted yet the default value is 0.

votersId: array composed by the identifiers of HEIs who already voted.

Consortium smart contracts also store a threshold value, representing the minimum number of votes necessary to confirm the inclusion of a new HEI. If this value is reached, the contract will assign the institution's DID to the corresponding smart contract address by storing this association in the contracts mapping. When a voting process finishes, the respective element in the registerPolls needs to be removed. However, the voters mapping needs to be explicitly reset to default values, but it is not possible to iterate through a mapping in Solidity. Therefore, the votersId array is used to retrieve every HEI identifier that voted in the poll, and reset each respective entry in the voters mapping. Finally, this registerPolls element can be removed, with all the attributes within being set to default.

However, before the voting system can be used to assure complete governance decentralization, there needs to be an initial \founder" group of HEIs to start a voting process. Therefore, this project defined three as the minimum number of HEIs needed to activate the contract's voting system. As stated previously, there is an entity that jumpstarts a Consortium smart contract by deploying it to the blockchain. Additionally, this entity will also have access to a specific function, registerFounderHEI( ), that allows it to register new HEIs on the contract. This function can only be called by the defined entity, and whenever the number of registered HEIs is equal to three, this entity loses all control over the contract and is blocked from invoking the registerFounderHEI( ) function. Henceforth all decisions taken on the contract need to be approved through the decentralized voting system.

If there is the need to remove a HEI from the consortium, the remaining HEIs can approve it by invoking the cancelHEI( ) function with the respective HEI identifier as a parameter. Similarly to registerHEI( ), if this function was called for the first time for a distinct HEI identifier, a new element is added to the cancelPolls mapping. This mapping links the identifier of a HEI, currently being voted to be removed from the consortium, with a CancelVote structure. For this operation there is no need to store the value of a contract address. Therefore, the only difference between this type of structure and a HEIVote structure, is the absence of this field. If a cancellation poll is successful, the association between the HEI's DID and its respective smart contract address is excluded from the contracts mapping. When a voting process finishes, the respective element in the cancelPolls needs to be removed. Therefore, similarly to the register polls, the votersId array is used to retrieve every HEI identifier that voted in the poll, and reset each respective entry in the voters mapping. The respective cancelPolls element is then removed, with all the attributes within being set to default.

Voting system threshold

Since the number of members of a consortium is dynamic, its is obvious that the threshold should not be a static value. Therefore, a Consortium smart contract needs to provide a decentralized option to alter this value. This was accomplished by implementing a changeThreshold( ) function. When the consortium decides that it needs to adjust the threshold value, it starts a voting process where members can vote by invoking the changeThreshold( ) function with the new value as a parameter. This operation instantiates a new ThresholdVote structure, similar to the HEIVote structure with the contractAddress field being replaced by an integer representing the new threshold value. If the number of votes reaches the old threshold value, the latter will be replaced with the new one.

This functionality gives even more emphasis on the importance of decentralization for this project, since it provides the possibility for the consortium to adjust the threshold value to its needs. If the value was always static or only modified by a centralized entity, this system would assure a pseudo-decentralization, which in reality means that a defined set of members would always have control over the decisions approved for contract alterations.

4.3.3 Higher Education Smart Contract

The HE smart contract can have different versions depending on different design options, according to the functionality that needs to be provided. However, a single design is presented in this document, represented in Figure 4.7, that encapsulates all the essential requirements for a fully functional ecosystem.

Whenever a new HEI decides to join this platform, it needs to create its own Ethereum account. This can be accomplished by running the createAccount script, contained in the Higher Education App module, that will return the new account's address and the respective private key. Afterwards, it needs to deploy its own smart contract on the Ethereum blockchain by executing the deployContract script (also available in the HEApp module), and communicate to the consortium that it desires to enter. Every HE smart contract provides three essential operations:

registerCertificate( ) : register a certificate on the blockchain by storing the cryptographic hash and metadata received as arguments.

revokeCertificate( ) : revoke a certificate by identifying it through the cryptographic hash and metadata received as arguments.

verifyCertificate( ) : verify if a specific certificate is registered on the contract, and returns

the respective result to the function caller.

The functions registerCerti cate() and revokeCerti cate() are restricted to only being called by the Ethereum account of the institution that deployed the smart contract. Additionally, this contract de nes a certi cates mapping that pairs a student's civil id with the cryptographic hash generated from his earned education credential.

In this project's design phase, there were two possible implementations for revoking certi - cates:

1. When calling revokeCerti cate() , the function veri es if the certi cate is in the registered list, and deletes it in case it is.

2. The smart contract holds a certi cate revocation list (CRL), and every time revokeCer- ti cate() is called for a certi cate to be revoked, this certi cate's relevant information is added to the CRL. When verifyCerti cate() is called, it needs to verify if the speci ed certi cate is contained in the registered list, but also if it is not present in the CRL.

The rst implementation has the advantage of occupying less memory, since it only holds a single list and every revoked certi cate gets removed from the list. The second implementation keeps a revocation history, that makes it easier for the certi cate owners to know when their certi cate was revoked. However, this approach is more computationally costly, as for each veri cation there is the need to search in two di erent lists, registered and CRL.

The rst solution turned out to be chosen for the project since it was the least computation- ally costly, not only when registering a certi cate, but also when verifying it due to the usage of a singular list.

4.3.4 Higher Education Application

As illustrated in Figure 4.8, when a student graduates, a HEI sta member interacts with its Academic Management System (AMS) to request the student's education certi cate. The AMS will then generate the certi cate in PDF format and make it available for the sta , that typically prints it.

This system was developed to be integrated in a seamless way with any AMSs. Instead of printing the certi cates, the latter are stored in a folder located in the HEI's servers. That folder, named registered_certi cates , is shared with the Higher Education App (HEApp), that detects whenever a new certi cate le is added to the folder and automatically registers the certi cate in the blockchain.

HEApp is written in JavaScript, and provides three main functionalities:

Figure 4.8: Certi cate Registration operation.

registerCerti cate( ): receives a path to a certi cate to be registered in the blockchain.

revokeCerti cate( ): receives a path to a certi cate to be revoked in the blockchain.

registerIPFS( ): receives a certi cate to be stored and shared in IPFS.

On startup, the HEApp deploys a watcher tracking the shared folder's content. Whenever a new education certi cate is stored in the registered_certi cates folder, the watcher throws an event that will be handled by the registerCerti cate( ) function. The latter receives the le path as a parameter and retrieves the respective certi cate and student's civil id. Afterwards, the hashing algorithm SHA-256 is used to obtain the certi cate's cryptographic hash that will be attached to a new Ethereum transaction. When creating a transaction, aData eld is de ned with two parameters: Civil id and certi cate hash. The transaction will be signed with the HEI's private key and sent to the Ethereum node, which will commit it to the blockchain network. When processing this transaction, the network will execute theregisterCerti cate( ) function from the HEI's smart contract. This will result in a state change on the certi cates mapping, since it will associate the civil id key entry to the committed cryptographic hash value.

After the transaction is con rmed, the registerIPFS( ) function is invoked and it receives as parameters the le name and its content in the form of bytes. Afterwards, these parameters are sent to the IPFS node, running on the HEI's servers, which will then store and share the le with the peer-to-peer network. Lastly, this operation will return a multihash value, that can be used to retrieve this le by anyone connected to the IPFS network.

When there is the need to revoke a certi cate registered on the blockchain, the HEI interacts with its respective AMS, that adds the certi cate to be revoked to another shared folder named revoked_certi cates. Since the HEApp deployed a second watcher that tracks this folder, the

revokeCerti cate( ) function is called to handle this request. The latter will receive a path to the le and attach the student's civil id to a Data eld on a new transaction. This transaction is signed with the HEI's private key and sent to the running Ethereum node that will submit the transaction to be approved by the blockchain network. Lastly, the transaction is processed by executing the revokeCerti cate( ) function from the HEI's contract.

4.3.5 Recruiting Application

Figure 4.9: Recruiting App interface.

Whenever an organization opens a new job position, there is a need to make a selection of the most appropriate candidates. After this selection is completed, all education certi cates from the chosen candidates need to be veri ed for their legitimacy. For employers to receive a candidate's education certi cate, there are two possible approaches that can be taken:

1. Receive the complete education certi cate from the candidate.
2. Receive the IPFS multihash corresponding to the candidate's education certi cate and retrieve the complete certi cate through the IPFS node running on the recruiting organization's servers.

After acquiring the complete certi cate, the employer can now interact with the Recruiting App interface (Figure 4.9) to verify its authenticity and integrity. The employer will start by inserting, in the respective elds, the candidate's civil id and the DID corresponding to the Higher Education Institution that issued his certi cate. Afterwards, the employer uploads the certi cate le to the application and completes the request for its veri cation.

As represented in Figure 4.10, the Recruiting App processes this request by communicating with the Ethereum node (running on the recruiting organization's servers) to call the getHEI()

function on the Consortium Smart Contract. This function receives the HEI's DID as a parameter and returns the respective contract address. Next, the contract associated with the returned address is called through theverifyCerti cate( ) function, with the candidate's civil id as a parameter. This function will return the cryptographic hash corresponding to the submitted civil id. The Recruting App will then utilize the SHA-256 hashing function on the certi cate le received from the candidate. The result of this operation will then be compared with the hash received from the contract and if the hashes match, the veri cation was successful and, consequently, the education certi cate is authentic.

Figure 4.10: Certi cate Veri cation operation.

It is important to distinguish between read-only and state-changing smart contract functions. A function that can potentially change the state needs to be invoked through a transaction, that is sent to the network for veri cation. The sender needs to wait for this transaction to be mined, causing a signi cant delay until it is con rmed. On the other hnd, read-only functions are marked with the keyword \view" representing the promise to not modify the state. Since each node has a copy of the state, the latter can be analyzed without assistance from the Ethereum network.

The verifyCerti cate( ) function available on the HE contract is a read-only function. Therefore, it is executed on the local node, running much faster than transactions that require network veri cation. Since it is executed locally, every veri cation is free which is extremely bene cial for any recruiting organization.

4.3.6 Consortium Application

The process of accepting/dismissing HEIs from a consortium needs to be dynamic and reliable. This system was developed for members to have complete trust in it, therefore, transparency is another major property that needs to be assured. Blockchain technology provides this property, being one of the main bene ts of adopting such a disruptive system. TheConsortium smart contract is the core component of the voting system, where all polls and the respective votes are registered. Therefore, there is a complete transparency of the voting process, since the votes of all HEIs are openly available on the blockchain.

An interface was also developed to support theConsortium App, that will facilitate the interaction of members with the consortium contract. This interface provides three forms and lists of ongoing polls, one for each of the contract's main voting operations.

Register HEI

As stated previously, whenever a HEI decides to join the platform, it deploys its own HEI smart contract on the Ethereum blockchain and communicates its intents to the consortium. After receiving this information, one of the consortium's members can start a new poll by interacting with the \Register HEI" form (Figure 4.11) in the Consortium App interface. The \HEI Identi er" eld receives the DID associated with the HEI wanting to join the consortium, while the \Contract address" eld corresponds to the address of the newly deployed HEI contract.

After one of the members enters the correct values and presses the \submit" button, the Consortium App will create a new transaction and sign it with the private key of the HEI that is using the application. This transaction will invoke the registerHEI( ) function in the Consortium smart contract, with the inserted values as arguments, initiating a new poll by adding a new element to the registerPolls mapping. As explained in Section 4.3.2, this new entry associates the HEI identi er with a HEIVote structure, that will de ne the following attributes:

contractAddress is set to the value received as an argument.

positiveVoteCounter is set to 1, since the vote to start a new poll is always a positive vote.

negativeVoteCounter is equal to 0.

Insert a new entry in the voters mapping, associating the voting HEI's identi er with the value 1 (positive vote).

Append the voting HEI's identi er to the votersId array.

When another HEI runs its own Consortium App, it will start by querying the consortium contract for currently ongoing polls. This is accomplished by calling the getPollingInfo( ) function, that will return four different elements stored in the contract:

registerIdArray : array containing the identifiers associated with the HEIs currently being voted to join the consortium.

registerContractArray : array containing the HEI contract addresses associated with the HEIs currently being voted to join the consortium.

cancelIdArray : array containing the identifiers associated with the HEIs currently being voted to leave the consortium.

thresholdVotingValue: new value currently being voted to replace the current threshold.

Since it is not possible to iterate through the registerPolls mapping, the registerIdArray and registerContractArray data structures were created in the consortium contract. Every time a new register poll is started, the identifier of the HEI is appended to the registerIdArray and the respective contract's address added to the registerContractArray. After receiving both of these arrays through the getPollingInfo( ) function, the "Pending HEI registrations" list, available on the interface, is updated.

Figure 4.11: Register HEI form.

An example is represented in Figure 4.11, where previously one of the consortium's members interacted with the interface and started a new poll through the "Register HEI" form. When another member executed its Consortium App, this poll is now available for voting in the "Pending HEI registrations" list. The member can then use the positive or negative buttons to vote, which will create a new transaction and sign it with the member's private key. This transaction will be sent to the Ethereum blockchain network, that will process the consortium's

contract registerHEI( ) function with the HEI identi er, contract address and voting value (1 for positive, 2 for negative) as arguments.

Cancel HEI

Even though a HEI leaving the consortium will be a rare occurrence, this may happen due to a variety of reasons. It can be the HEI itself that decides to leave or the other members agreeing on its exclusion. If this is the case, one of the members (including the HEI itself) can start a new poll by interacting with the \Cancel HEI" form available on the Consortium App interface.

As displayed in Figure 4.12, the DID associated with HEI being voted o the consortium can be entered in the \HEI Identi er" eld. When submitted, a transaction is created and signed with the HEI's private key, and sent to the blockchain network. This transaction invokes the cancelHEI( ) function in the consortium contract, with the HEI identi er as an argument. A new cancellation poll is started and a CancelVote structure is instantiated with the following attributes:

positiveVoteCounter is set to 1, since the vote to start a new poll is always a positive vote.

negativeVoteCounter is equal to 0.

A new entry is added to the voters mapping, associating the voting HEI's identi er with the value 1 (positive vote).

The voting HEI's identi er is appended to the votersId array.

Similarly to the \Register HEI" operation, it is not possible to iterate through the cancelPolls mapping, therefore, the cancelIdArray was created in the consortium contract. If there is the necessity of excluding a member from the consortium, a new cancellation poll is initiated, and the identi er of the HEI being removed is appended to the cancelIdArray.

After a poll is started in the consortium contract, one of the others members can vote on it by interacting with Consortium App interface. On startup, the application will update the \Pending HEI cancellations" list by calling the getPollingInfo( ) function from the contract, with one of the returning values being the cancelIdArray. The DIDs stored in this array will then be presented in the interface, while giving the possibility to vote positively/negatively on the respective poll, as represented in Figure 4.12. When pressing one of the buttons, the cancelHEI( ) function is invoked by creating, signing and sending a transaction to the Ethereum blockchain network, with the HEI identi er and the voting value (1 or 2) as arguments.

Figure 4.12: Cancel HEI form.

Change threshold

For a poll to be successful/rejected, the number of positive/negative votes needs to reach the threshold value de ned in the consortium contract. However, this value needs to be dynamic to match the uctuation in the number of consortium members. Therefore, the Consortium App interface provides the \Change Threshold" form, displayed in Figure 4.13, which receives the new value to be proposed and starts a new poll to replace the threshold.

After having lled and submitted the \New value" eld, the application will create a new transaction signed with the private key of the HEI utilizing the interface. This transaction is sent to the blockchain network and will be processed by executing the changeThreshold( ) function of the consortium contract. The function receives the new threshold and the value 1 (positive vote) as arguments. When the transaction is con rmed by the network, a new ThresholdVote structure is instantiated in the consortium contract with the following attributes:

value corresponds to the new threshold that is being voted to replace the old value.

positiveVoteCounter is set to 1, since the vote to start a new poll is always a positive vote.

negativeVoteCounter is equal to 0.

A new entry is added to the voters mapping, associating the voting HEI's identi er with the value 1 (positive vote).

The voting HEI's identi er is appended to the votersId array.

Contrarily to the other voting operations, it was established that a consortium can only vote on a singular threshold poll at a time. Therefore, there is no need to a store an array for threshold

Figure 4.13: Change threshold form.

polls on the consortium contract. When another HEI starts its Consortium App, the getPolling-Info( ) function is called and a thresholdVotingValue is returned. The latter corresponds to the value currently being voted to be the new threshold. The \Pending threshold alteration" list is updated with this value, as represented in Figure 4.13. The positive and negative buttons can then be used to vote on this alteration, creating and signing a new transaction that invokes the changeThreshold( )function from the consortium contract. This function receives and processes the new threshold and the respective voting value.

HEI Contract Address

The Consortium App also provides an auxiliary functionality in addition to the main voting operations. The form \HEI Contract Address" available on the interface (Figure 4.14), allows any consortium member to retrieve the contract address associated with a given HEI identi er. A member inserts the DID linked with a HEI in the \HEI Identi er" eld and presses the \Submit" button, which will trigger a call from the application to the consortium contract. This request will call the getHEI( ) function from the contract, that receives a HEI identi er as an argument and returns the respective HEI contract address.

If the entered DID corresponds to a HEI that is part of the consortium and, therefore, registered in the consortium contract, the respective HEI contract address is displayed on the interface. Otherwise, it will present an error message indicating that the HEI is not a member of the consortium. This functionality is useful for members to check if a speci c HEI is registered on the contract, as represented in Figure 4.14.

Figure 4.14: HEI Contract Address form.

4.3.7 Alternative Designs

Before fully implementing the described platform, a few other design options were investigated. The main alternative for certi cate registration/revocation, represented in Figure 4.15, would be the use of services provided by an external company. These services o er the possibility for the AMS to invoke an API for certi cate registration/revocation, sending the education certi cate as an argument. The external company will then process and register the certi cate on the blockchain accordingly. This method would relief the burden of the need for additional internal resources from the higher education institution, but would result in a delegation of trust to the external company, in the proper processing of critical assets for the graduated students.

Figure 4.15: Certi cate Registration - alternative solution.

For certi cate veri cation another possible approach would allow for a straightforward and uncomplicated operation, represented in Figure 4.16, as there is no need for organizations to maintain any application and Ethereum node running on their servers. However, the chosen approach grants a procedure where the organization does not need to trust any third-party to retrieve information about a certi cate, since it does not need any intermediary applications to process the veri cation request. Furthermore, it also contributes to decentralization, one the main properties present in this project, since the external company would be a point of centralization for operations from di erent Higher Education Institutions and recruiting organizations.

Figure 4.16: Certi cate Veri cation - alternative solution.

4.4 Summary

This section started by giving an overview on the QualiChain project architecture. Afterwards, it was introduced the Recruitment in Public Sector Pilot use case to which this system was applied to. A generalized solution based on Ethereum blockchain technology o ering the ability to be integrated with any Academic Management System was then described. The two core smart contracts were explained, along with the three developed applications which interact with them. Finally, alternative designs for the system architecture were presented.

Chapter 5

Evaluation

This section will start by describing the implemented methodology, giving insight on di erent decisions that had to be made prior to the evaluation. It also de nes the di erent indicators that were measured to benchmark the system. Afterwards, the experimental results are presented for each module, discussing all the obtained results. Finally, an in-depth explanation of the previous results provides all the necessary information to conclude the system evaluation and the respective bene ts of adopting it.

5.1 Methodology

Ethereum is a permissioneless blockchain [Pec17, Cor19], meaning that anyone can join the network and utilize the services provided by its infrastructure. To rationalize the use of its resources, Ethereum charges Ether for the execution of smart contracts. Although this is an important feature to prevent attackers from requesting computationally expensive operations to slow down the network, it is inconvenient when designing and testing new applications. For that purpose, there are a set of Ethereum test networks or testnets.

Testnets o er a safe environment for testing and experimenting on projects before launching them on Ethereum's main network. There are three main test networks available, each one with their respective advantages:

Ropsten : the one that most resembles the main network because it uses a proof-of-work consensus algorithm, meaning that nodes are in charge of maintaining the network. Blocks are introduced at an average of 30 seconds.

RinkeBy : uses the consensus algorithm proof-of-authority, meaning that there is a pre-approved group of validators. The latter validates transactions and blocks within the

network and is usually composed by a small number of members, in order to ensure efficiency and secure manageability of the network. All Ether are already mined and only distributed on demand. Blocks are generated at intervals of 15 seconds.

Kovan : uses the same consensus algorithm as the Rinkeby network, which is a proof-of-authority. All Ether is also already mined, and its retrieval is done through a request. Blocks on the Kovan network are generated at intervals of 4 seconds.

Ropsten was the test network chosen, since it is the one that most resembles the main network by using a proof-of-work consensus algorithm, meaning that nodes are in charge of maintaining the network. Blocks are introduced at an average of 30 seconds [Boi14].

The developed system was evaluated by benchmarking the performance of each module. This benchmark was composed by three different indicators:

Throughput of the solution when there are a large number of requests being processed, indicating how many operations the system can process in a predefined time interval.

Cost of each certificate operation. Measuring the gas cost, and the respective Ether cost of a request from a client. This assessment is done by taking into account the computational cost needed to process the operation.

Latency of a certificate operation, indicating the delay between a client request and the system's response.

5.2 Experimental Results

This subsection describes and presents the evaluation results for each module, with a detailed explanation of all testing procedures. A modified version of each application was created to more easily obtain the desired performance metrics. This adaptation allows the system to print the processing time of each operation.

5.2.1 Higher Education Module

This module was developed to support HEIs in certificate management and sharing. There are two main operations provided by the Higher Education App: register certificate and revoke certificate. Both of them were evaluated and all steps taken will be described.

Register Certi cate

This operation gets triggered when a new education certi cate is added to the \registered certi cates" shared folder. The certi cate will then be processed by the HE App, and a transaction is sent to the Ethereum blockchain network. The registerCerti cate( ) function from the HEI contract is invoked and the certi cate's hash and metadata are registered.

Throughput represents the ratio between the number of certi cates to be registered by the amount of time needed to register those certi cates on the contract. A certi cate is considered to be registered when the corresponding transaction is included in a new Ethereum block and appended to the blockchain. This information can be obtained from the Etherscan Blockchain Explorer website.

This indicator was measured by repeating 30 times the described operation for 5 di erent sample sizes: 25, 50, 100, 250 and 500 certi cates. An example certi cate le of approximately 100 kB was used for all iterations.

(a) Execution time (b) Throughput

Figure 5.1: Register Certi cate throughput results.

Figure 5.1(a) displays the execution time for a de ned amount of certi cates to be registered. As expected this value rises as the amount of certi cates increases. The represented standard deviations correspond mostly to network delays, queue sizes and the Ether amount o ered to the miners for processing a transaction. On the other hand, Figure 5.1(b) represents the operation throughput for di erent certi cate sample sizes. It is important to verify that the processing and submission of each transaction by the application is just a minor portion of the full con rmation period, the majority corresponds to the mining of all the issued transactions.

Latency represents the amount of time needed to register a single certi cate on the HEI contract. This indicator was also measured by repeating 30 times the same operation for 5 di erent certi cate le sizes.

Figure 5.2: Register Certi cate latency results.

In Figure 5.2 it is possible to observe that the certi cate's size has little to no in uence in the latency, since mining the respective transaction and adding it to a new block occupies the largest percentage of the processing time. Furthermore, certi cate les are never of a size large enough to in uence signi cantly the hashing function, therefore, the application processing time will always be identical. The standard deviations correspond mostly to network delays, queue sizes and the Ether amount o ered to the miners for processing a transaction.

Indicator	Throughput (certi cates/second)	Cost (Ether)	Latency (seconds)
Register certi cate	2.305	0.00023792	38.206

Table 5.1: Register Certi cate averages table.

The cost de ned in Table 5.1 is a constant value representing the Gas price * Gas used in Ether. As explained in section 2.2, the Gas price expresses the fee the sender is willing to pay per computational step. Since the registerCerti cate( ) function provided by the HEI contract will always take the same amount of steps no matter what certi cate it is registering, the cost value is the same at all times for each Gas price.

Revoke Certi cate

When there is the need to revoke an education certi cate that was previously issued, the respective le is added to the \revoked_certi cates" shared folder. The certi cate will then be processed by the HE App, the respective civil id retrieved and a transaction is sent to the Ethereum blockchain network. Finally, the revokeCerti cate( ) function from the HEI contract is invoked with the student's identi er as an argument, and the corresponding certi cate is revoked.

Figure 5.3: Revoke Certi cate throughput results.

Throughput represents the ratio between the number of certi cates to be revoked by the amount of time needed to revoke those certi cates on the contract. This indicator was measured exactly as the Register Certi cate operation, repeating 30 times for 5 di erent sample sizes: 25, 50, 100, 250 and 500 certi cates. An example certi cate le of approximately 100 kB was also used for all iterations.

The obtained results are shown in Figure 5.3, representing the operation throughput for the di erent certi cate sample sizes. Since the application processing time is almost negligible compared to the amount of time it takes a transaction to be con rmed, the throughput will rise when increasing the number of transactions waiting to be included in a new block, considering that blocks are introduced at a consistent rate.

Figure 5.4: Revoke Certi cate latency results.

Figure 5.4 illustrates the obtained results from measuring the amount of time needed to revoke a single certi cate on the HEI contract. This indicator was measured by repeating 30 times the same operation for 5 di erent certi cate le sizes. Once again, it is evident that the

certificate file size has little influence in the latency. This value has a high degree of variation since it depends on how congested the Ethereum network is at the time.

As shown in Table 5.2, the cost value is inferior to the Register Certificate operation. The amount of computational effort that it takes to execute a revocation is lower than a registration, since the revokeCertificate() function on the contract only resets to the default value the corresponding certificate's mapping entry. While the registerCertificate() function assigns and stores a new certificate's hash value on the mapping, resulting in a higher transaction fee.

Indicator	Throughput (certificates/second)	Cost (Ether)	Latency (seconds)
Revoke certificate	2.220	0.00023215	38.627

Table 5.2: Revoke Certificate averages table.

5.2.2 Recruiting Module

Recruiting organizations need to assure the authenticity of their candidates' education certificates. Therefore, this module was developed for recruiters to be able to easily verify any certificate through an application interface by uploading the received certificate file. The application will start by calling the consortium contract to retrieve the respective HEI's contract address. TheverifyCertificate() function from this contract is then invoked, and the returned hash is compared with the hash from the uploaded certificate file.

Although the interface was developed with the intent to only verify a single certificate at a time, to evaluate the throughput it was necessary to implement the possibility to verify multiple certificates simultaneously. This modification to the original Recruiting App receives a set of certificate files and prints the amount of time it took to verify them.

Figure 5.5: Verify Certificate throughput results.

As represented in Figure 5.5, the throughput was measured with di erent certi cate sample sizes. Taking the example of verifying 500 certi cates for a job position, being able to check the authenticity of approximately 160 certi cates per second is an extremely faster method than any veri cation that recruiters are performing currently.

Figure 5.6: Verify Certi cate latency results.

It was veri ed that the certi cate le size does not greatly in uence the latency, as shown in Figure 5.6, since certi cate les will never be large enough to have a major impact on the performance of the application's hashing function. For example, a 250 kB certi cate le can be proved authentic in just approximately 0.91 seconds, a notable improvement on any other veri cation methods.

One of the biggest factors for why an organization should join this platform is the cost of each veri cation. As presented in Table 5.3, it is free to verify a certi cate's authenticity through the developed application interface. A read-only call simply requests the local Ethereum node to process theverifyCerti cate( ) function from the contract and returns the respective result. Therefore, there is no need to interact with the Ethereum blockchain network, thus no transaction is created and, consequently, this operation is free of charge. This is only possible because the local node keeps a copy of the Ethereum state, being able to execute any function that is not potentially state-changing.

Indicator	Throughput (certi cates/second)	Cost (Ether)	Latency (seconds)
Verify certi cate	90.242	0	0.915

Table 5.3: Verify Certi cate averages table.

5.2.3 Consortium Module

The Consortium App interface was developed for members to vote on all consortium related decisions, such as a new member joining, a member being removed or altering the threshold value. Contrarily to the other modules the application throughput was not measured. Considering that each vote should be carefully considered and well thought out, the application is not expected to send multiple voting transactions at once to the consortium contract. Therefore, there is no need to measure the amount of votes per second that the application can handle.

Register HEI

When a new HEI makes a request to join the consortium, the members can vote if they approve or not this decision by interacting with the developed interface. Each vote will create a new transaction that invokes the registerHEI( ) function from the consortium contract, instantiating a new register poll.

As previously explained in section 4.3.6, the interface provides a \Register HEI" form where one of the members inserts relevant information about the HEI trying to join the consortium. This will start a new poll on the consortium contract, that will then be displayed on the pending registration list available on the other members' interface. Although both procedures invoke the sameregisterHEI( ) function, starting a new poll through the \Register HEI" form takes a di erent amount of computational steps than simply voting through the pending list. Therefore, the latency and the respective cost were measured and analyzed for both methods.

Figure 5.7: Register HEI latency results.

For this testing purposes, it was necessary to create a new Ethereum account corresponding to the HEI requesting to join the consortium, along with the deployment of an HEI contract. This procedure is obligatory since the network checks if the account and the respective contract

are actually part of the Ethereum blockchain. Otherwise, the transaction is rolled back and the speci c error is returned.

Figure 5.7 represents the obtained results that show a slightly higher average latency for the poll starting vote than for a regular vote through the pending list. Although there is a minor di erence in the averages, the standard deviation de nes an ample range of latency values in both methods. This phenomenon is caused by the high uctuation on the amount of time needed for a transaction to be mined, since it depends on a wide variety of factors.

The number of computational steps required to execute a transaction that initiates a new poll is higher than a regular vote, therefore, the corresponding fee will also be larger. While the rst operation instantiates a HEIVote structure that is used to store all the information relative to the new poll, the second only updates this already created structure. For that reason, the amount of gas spent and the proportional Ether cost will always be higher.

Indicator	Latency (seconds)	Cost (Ether)
Starting vote	48.482	0.00216189
Vote	40.418	0.00099481

Table 5.4: Register HEI averages table.

Cancel HEI

Although it is expected to be a rare occurrence, a HEI might need to leave the consortium or even the other members themselves believe that this HEI should no longer be allowed to be part of the consortium. When this is the case, the members can vote if they approve or not this decision by interacting with the developed interface.

Figure 5.8: Cancel HEI latency results.

As explained in section 4.3.6, a member can start a new poll to remove a HEI from the consortium through the \Cancel HEI" form. A transaction is created and sent to the Ethereum network invoking the cancelHEI( ) function from the consortium contract, receiving the inserted HEI identi er as an argument. Similarly to the previous operation, other members will then be able to vote for the ongoing poll through the cancellation pending list also available on the interface. The latency and respective cost of both methods were measured.

The obtained results represented in Figure 5.8, substantiate the in uence of network congestion on the operation's latency. Although the register and cancel operations are very similar, the experiments were performed at di erent dates, resulting in a variety of factors changing. This is the reason for the lower latency when comparing with the register voting operation. These factors will be further explained in the next section.

Indicator	Latency (seconds)	Cost (Ether)
Starting vote	26.574	0.00153014
Vote	30.180	0.00098089

Table 5.5: Cancel HEI averages table.

As expected, voting to start a new cancellation poll has a higher computational cost than a standard vote using the corresponding pending list. The cost values represented in Table 5.5 are lower than the Register HEI operation for both voting methods due to a lower number of computational steps taken, since a register poll also needs to store the HEI's contract address.

Change threshold

The consortium needs to adjust the threshold value according to changes made to the number of members or even the trust levels between them. Members can vote on this decision through the developed interface, starting a new poll by interacting with the \Change Threshold" form. A transaction is created that invokes the changeThreshold( ) function available on the consortium contract, initializing a new ThresholdVote structure that stores the proposed threshold value and all the related poll information. Other members can then vote if they approve or disapprove the new value through the pending list available on the interface.

Figure 5.9 represents the measured latency for both voting methods. Once more it is possible to observe a signi cant standard deviation value, depicting how variable is the time needed for a transaction to be mined and con rmed. The cost associated with a starting vote (Table 5.6) is lower than the other voting operations, since it was de ned that there can only be one ongoing threshold poll at a time. Therefore, there is no need to manage arrays like the other operations,

Figure 5.9: Change threshold latency results.

resulting in a reduced number of computational steps. Consequently, a regular vote has also a lower cost.

Indicator	Latency (seconds)	Cost (Ether)
Starting vote	36.187	0.00151218
Vote	32.074	0.00097918

Table 5.6: Change threshold averages table.

5.3 Result Analysis

To understand the obtained latency results it is necessary to comprehend how the Ethereum network operates. It is constituted by a large variety of nodes with a wide range of purposes on the network, as represented in Figure 5.10. When a new transaction is created by any of the evaluated applications, it is submitted to the local Ethereum node where it is validated to check if the signature corresponds to the account from which is being sent. Afterwards, the local node broadcasts the transaction to its adjacent nodes which will start a recursive broadcast process where every node that receives the transaction will broadcast it to others.

Miner nodes accept a transaction and add it to a pool, which groups all the received pending transactions. Since miners want to maximize their fees, this poll is sorted by gas price, resulting in transactions with higher gas prices being more likely to be included in the next block and appended to the blockchain. When a node starts a mining process, it can only chose a limited amount of transactions from the poll to add to the new block. This amount is a predefined Ethereum value called block gas limit, enforcing that the sum of the gas limits from every chosen transaction does not exceed this value. After selecting the transactions from the pool,

Figure 5.10: Example of an Ethereum network segment.

they are validated, executed and added to a new pending block. Afterwards, a competition starts among miners to solve a mathematical puzzle (proof-of-work), in which the first miner to solve the problem wins the privilege to add the new block to the Ethereum blockchain and collect all the fees paid by the transactions in the block. The miner then broadcasts the newly solved block to the network and upon receiving it, other nodes execute the transactions within the block and update their copy of the ledger accordingly.

As represented in Figure 5.10, Etherscan provides a few dedicated nodes connected to the Etherscan Blockchain Explorer website. As explained above, when a newly mined block is added to the blockchain, this dedicated nodes will also receive it and update their copy of the ledger. Through an API this website retrieves the requested information about a transaction from the provided nodes. This was the tool used as the source to calculate the amount of time it takes for a transaction to be confirmed and the respective fees.

After having explained how the Ethereum network operates, it is now possible to understand the mains factors that influence the latency of each operation. The time it takes for a transaction to be confirmed is then dependent on the offered gas price and how congested the network is at the time. Since there is a limited space in a block, the greater the demand to include a transaction in a new block, the higher the average gas price will be. Transactions offering higher gas prices will take priority, slowing down the confirmation of the other pending transactions. Therefore, network congestion is the biggest factor when it comes to the latency of an operation.

It is important to note that the same gas price was used for all the previously presented evaluations. This value was 10 Gwei which is equivalent to 0.00000001 Ether. Additionally, another evaluation was performed to examine how changing the offered gas price can influence the latency of the system's operations. Modifications were made so that before creating every trans-

Figure 5.11: Comparison between the latency of operations o ering di erent gas prices.

action, the application invokes the web3.eth.getGasPrice( )function from the web3.js library, requesting the last few blocks median gas price. For the register/revoke certi cate operations, a default 100 kB certi cate le was used. For simplicity, all consortium related operations only measured the latency for poll starting votes.

Figure 5.11 illustrates the di erence between the previously measured latencies always using a 10 Gwei default gas price value, and the latencies of the same operations while o ering the average gas price at the time. The average gas price returned was mostly 1 Gwei, which is a tenth cheaper than the rst value used. As expected, o ering a higher gas price resulted in a lower latency, however, the di erence between them is minimal. There is a clear exception on the Register HEI voting operation, which certainly resulted from a highly signi cant network congestion at the moment of the evaluation.

The latency of all the developed operations that handle transactions can be quite variable, although it is evident that the bene ts of adopting this system largely out-weight the minor waiting time for an operation to be completed. However, it is crucial to remember that all previously presented evaluations were performed on the Ropsten testnet It is obvious that the transaction frequency on the main network greatly increases, but miners now have an economic incentive to earn real coins which leads to bigger investments in computational power for transaction mining. Therefore, the results presented in this document are only an approximation of the real values, which can vary due to a much higher number of stakeholders.

5.4 Summary

This section started by describing the methodology that was used to evaluate the developed system. The di erent indicators that were measured to benchmark the system were also de ned.

Afterwards, the experimental results were presented for each module, discussing all the obtained results. Finally, an in-depth explanation of the previous results provided all the necessary insight to assess the bene ts and constraints of adopting this system.

Chapter 6

Conclusions

This document presents a solution for scepticism from employers when receiving candidate's education and employment credentials, while making the recruitment process more agile and e cient for organizations.

The de ned solution demonstrates an architecture that makes use of blockchain technology and integrates it with any Academic Management System. With this integration, an AMS can e ciently register a education certi cate on the blockchain, that would be easily veri able by any interested recruiting organization. Ethereum was selected as being the blockchain infrastructure more suitable for the project, with smart contracts having a crucial role on the solution's functionality. Three applications were developed for interacting with the platform. The Higher Education App enables HEIs to register/revoke their graduated students' education certi cates on the blockchain. The hash of each certi cate is stored in a mapping on the HE smart contract deployed by the respective HEI. The Recruiting App allows recruiting organizations to easily verify the authenticity of a candidate's education certi cate. Finally, the Consortium App offers the possibility for consortium members to vote on all consortium related decisions, such as determining if a new HEI should join. Members can vote by invoking the Consortium smart contract that provides di erent functions for distinct voting processes.

To evaluate the developed system, several application level Key Performance Indicators were measured. The solution was deployed on an Ethereum Test Network, where the speci ed KPIs were assessed. Although the obtained results show a relatively high uctuation on the latency of each operation mainly due to network congestion, the measured values are still unequivocally bene cial for any HEI or recruiting organization that adopts this system.

The solution is characterized throughout the document. Starting by presenting several background and related work, that supported the development of the project. Afterwards, the system architecture is presented specifying all the developed applications and smart contracts. Lastly,

an in-depth explanation on how the developed system was evaluated is then detailed.

In conclusion, the described solution was initially delineated to be strictly developed for a real use case of Recruitment in the Public Sector Pilot, integrating the FenixEdu platform and also the public institute in the elds of administrative modernization, simpli cation and digital administration. However, it was evident that developing a solution to integrate any AMS and recruiting organization was the best approach.

Part of this work was subject to scienti c evaluation and review through the article \Blockchain Ecosystem for Veri able Quali cations", Diogo Serranito, Andre Vasconcelos, Sergio Guerreiro, Miguel Correia, in Proceedings of the 2nd Conference on Blockchain Research Applications for Innovative Networks and Services - BRAINS 2020, September 28 { 30, 2020, Paris, France.

6.1 Future Work

The work accomplished by this research is expected to be a strong baseline from where other education institutions and employing organizations can adhere and build upon. There are many functionalties that can be added to supplement this platform, many of which were previously described for the QualiChain project (Section 4.1).

Adding support for di erent types of certi cates is a clear improvement that could be easily addressed in future developments. Higher Education Institutions award distinct quali cations for di erent educational courses. Being able to distinguish between them when registering on the blockchain would give the possibility for students to have a de ned portfolio on the HEIs contract.

Furthermore, providing students with the possibility to manage their own portfolio would be another interesting increment to the platform. If students have di erent education certi cates across a variety of HEIs, they might want to display or hide certain ones for an interested organization. This would give students more control over all their awarded education assets.

Bibliography

[AF18] Rodelio Arenas and Proceso Fernandez. Credenceledger: A permissioned blockchain for veri able academic credentials. In IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC) , pages 1{6, 2018.

[ARF+19] Merlinda Andoni, Valentin Robu, David Flynn, Simone Abram, Dale Geach, David Jenkins, Peter McCallum, and Andrew Peacock. Blockchain technology in the energy sector: A systematic review of challenges and opportunities. Renewable and Sustainable Energy Reviews, 100:143{174, 2019.

[Ben14] Juan Benet. Ipfs - content addressed, versioned, P2P le system. arXiv preprint arXiv:1407.3561, 2014.

[Boi14] Francis Boily. Explaining Ethereum test networks and all their di erences. https://medium.com/coinmonks/ethereum-test-networks-69a5463789be , 2018-05-14.

[Cac16] Christian Cachin. Architecture of the hyperledger blockchain fabric. In Workshop on Distributed Cryptocurrencies and Consensus Ledgers, 2016.

[Car] CareerBuilder. Fifty-eight percent of employers have caught a lie on a resume, according to a new CareerBuilder survey. https://www.careerbuilder.com/share/aboutus/pressreleasesdetail.aspx?sd=8%2F7%2F2014&id=pr837&ed=12%2F31%2F2014

[Cas18] A Castor. Cardano blockchain's rst use case: proof of university diplomas in greece. Bitcoin Magazine, January 2018.

[CD16] Konstantinos Christidis and Michael Devetsikiotis. Blockchains and smart contracts for the internet of things. IEEE Access, 4:2292{2303, 2016.

[Cor19] Miguel Correia. From byzantine consensus to blockchain consensus. Essentials of Blockchain Technology, chapter 3. CRC Press, 2019.

[DT17] Elizabeth Durant and A Trachy. Digital diploma debuts at mit. http://news.mit.edu/2017/mit-debuts-secure-digital-diploma-using-bitcoin-blockchain-technology-1017 2017.

[Ell09] Ryann K Ellis. Field guide to learning management systems. ASTD learning circuits, pages 1{8, 2009.

[Eth14] Ethereum team. Ethereum: A next-generation smart contract and decentralized application platform. White Paper, 2014.

[Eth19] Ethereum. Solidity. https://solidity.readthedocs.io/en/latest/index.html , 2016-2019.

[Gar15] J Garzik. Public versus private blockchains - part 1: Permissioned blockchains, part 2: Permissionless blockchains. BitFury, 2015.

[GC17] Alexander Grech and Anthony F Camilleri. Blockchain in education, 2017. Luxembourg: Publications O ce of the European Union.

[Gre15] Gideon Greenspan. Multichain private blockchain - white paper. http://www.multichain.com/download/MultiChain-White-Paper.pdf , 2015.

[Gro14] Object Management Group®. Business Process Model and Notation. https://pubs.opengroup.org/architecture/archimate3-doc , 2014.

[Gro17] Object Management Group. OMG® Uni ed Modeling Language® (OMG UML®). https://www.omg.org/spec/UML/ , 2017.

[IT00] ITU-T. International Telecommunication Union. ITU-T Recommendation X.509: The Directory: Public-Key and Attribute Certi cate Frameworks, 2000.

[KGDS18] Sukrit Kalra, Seep Goel, Mohan Dhawan, and Subodh Sharma. Zeus: Analyzing safety of smart contracts. In NDSS, pages 1{12, 2018.

[Met16] Matthias Mettler. Blockchain technology in healthcare: The revolution starts here. In IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom) pages 1{3. IEEE, 2016.

[Nak08] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008.

[Pec17] Morgen E Peck. Blockchains: How they work and why they'll change the world. IEEE Spectrum, 54(10):26{35, 2017.

[Qua18] QualiChain consortium. Qualichain - decentralised quali cations' veri cation and management for learner empowerment, education reengineering and public sector transformation. Projecto proposal, mar 2018.

[RSA78] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120{126, 1978.

[S+11] Antonio Rito Silva et al. The FenixEdu project: an open-source academic information platform. Instituto Superior Tecnico, https://ciist.ist.utl.pt/projectos/Fenix.pdf, mar 2011.

[Sta] StatisticBrain. Resume Falsi cation Statistics. https://www.statisticbrain.com/resume-falsification-statistics.

[The19] TheOpenGroup. ArchiMate R 3.1 Speci cation. https://pubs.opengroup.org/architecture/archimate3-doc, 2019.

[Tsu92] Gene Tsudik. Message authentication with one-way hash functions. *ACM Computer Communications Review*, 22(5):29{38, October 1992.

[UK 16] UK Government O ce for Science. Distributed ledger technology: beyond block chain. A report by the UK Government Chief Scienti c Adviser, 2016.

[Und16] Sarah Underwood. Blockchain beyond Bitcoin. *Communications of the ACM*, 59(11):15{17, 2016.

[Vra17] Harald Vranken. Sustainability of bitcoin and blockchains. *Current opinion in environmental sustainability*, 28:1{9, 2017.

[W3C] W3C. Decentralized Identi ers (DIDs) v1.0. https://www.w3.org/TR/did-core/.

[XPZ+16] Xiwei Xu, Cesare Pautasso, Liming Zhu, Vincent Gramoli, Alexander Ponomarev, An Binh Tran, and Shiping Chen. The blockchain as a software connector. In *13th Working IEEE/IFIP Conference on Software Architecture*, pages 182{191, 2016.

[ZL19] Shijie Zhang and Jong-Hyouk Lee. Analysis of the main consensus protocols of blockchain. *ICT Express*, 2019.

[ZS18] Kaspars Z le and Renate Strazdina. Blockchain use cases and their feasibility. *Applied Computer Systems*, 23(1):12{20, 2018.