SCAR: Smart Contract Academic Registry

A Blockchain approach for Academic Registry

Diogo Rodrigues
49513@alunos.isel.pt

Gonçalo Frutuoso
49495@alunos.isel.pt

Supervisors

Cátia Vaz, ISEL
cvaz@cc.isel.ipl.pt

Alexandre Francisco, IST
aplf@tecnico.pt

Final Report written for Project and Seminary
Licenciatura em Engenharia Informática e de Computadores

June 2, 2024

Abstract

This is the abstract.

Resumo

Aqui fica o resumo.

Acknowledgments

Here goes the acknowledgments.

Index

List of Figures

List of Tables

Chapter 1

Introduction

This is where the introduction goes to.

1.1 Outline

This is where the outline goes to. The outline is a guide for the reader to understand the structure of the document.

Chapter 2

Background

Nowadays, the authenticity and accessibility of academic certificates play a crucial role in ensuring trust and credibility in various domains, ranging from education to employment and beyond. The current and traditional *paper-based* system of issuing and verifying academic certificates is not only time consuming but also prone to a lot of fraud and manipulation. The widespread issue of counterfeit certificates[21, 25], coupled with inefficient verification processes and the risk of loss or damage, highlight the need for a more reliable and secure academic certificate registry system.

The current system of academic certificate registry faces numerous challenges. Firstly, the reliance on paper-based is problematic. Paper-based certificates are easily forged and tampered with. This undermines the credibility and integrity of academic qualifications. Secondly, the manual verification process is time-consuming and prone to errors, leading to delays in credential validation, possible fraudulent activities and also potential loss of revenue for institutions due to errors in the manual release. Thirdly, since the issue of certificates from educational institutions are mostly centralized, this intensify the difficulty of maintaining a unified and updated registry, avoiding efficient verification mechanisms.

2.1 Overview

There are several approaches to address the challenges of the traditional academic certificate registry system. One such solution is the implementation of centralized databases managed by government or regulatory authorities, where educational institutions are required to submit digital copies of certificates for verification purposes [34]. Other solution is the adoption of distributed systems where the data is stored across multiple nodes in a network, but not all nodes have the same equal authority and the data is not fully decentralized. Which means that there's an entity that has control over the network [31]. This solution is not centralized neither fully distributed but have a mix of both approaches that makes the system more secure and reliable than only a centralized one. The third solution that we will approach is a fully distributed one that offers a decentralized, secure and tamper-proof ledger where certificates can be stored and verified where there is no single entity that has control over the network. This solution is based on the *blockchain technology* [7, 29].

2.2 Centralized Systems

Several attempts have been made to address the challenges of the traditional academic certificate registry system. How we can see in the Figure 2.2, one such solution is the implementation of *centralized databases* [24] managed by government or regulatory authorities, where educational institutions are required to submit digital copies of certificates for verification purposes. Although this approach aims to centralize certificate records and simplify the verification process, it still faces challenges such as the risk and concerns of data privacy and security, interoperability issues between different databases and the need of a trusted third party to manage the database. This centralized systems often only store grades and not the actual certificates which undermines the credibility and integrity of academic qualifications. Moreover, the reliance on a central authority to manage the registry increases the risk of fraud and manipulation, as the data can be altered or deleted by a single entity.

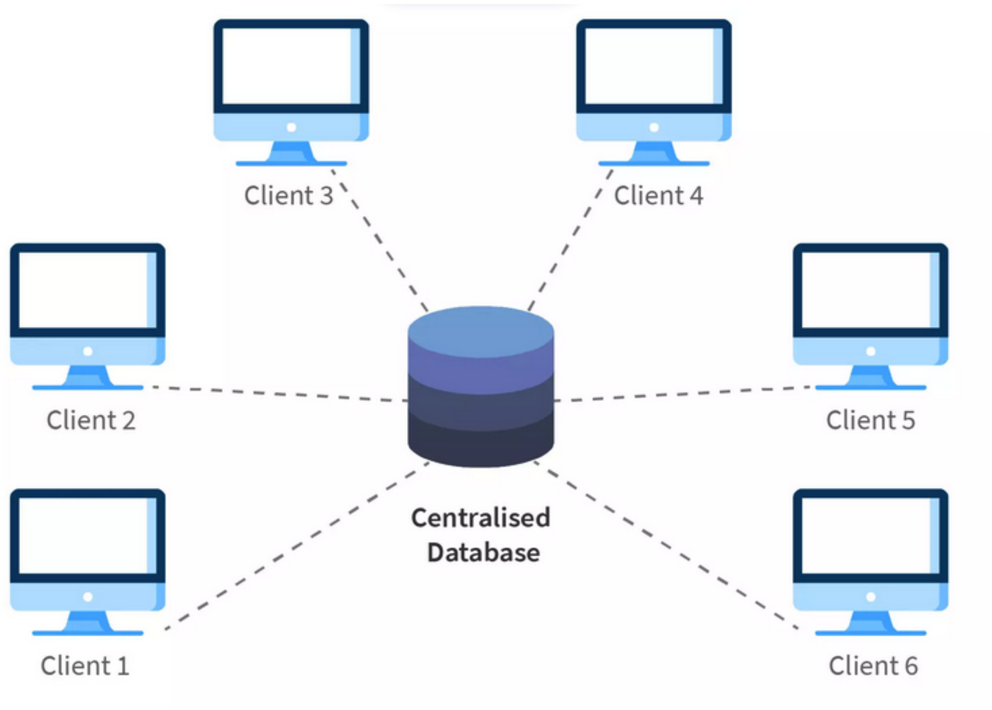

Figure 2.1: Centralized System [30].

2.3 Distributed Systems

Another solution to address the challenges of the traditional academic certificate registry system could be the adoption of distributed systems where the data is stores in multiple nodes in a network. Has we see in the Figure 2.3, different that what we've seen in the previous section, the data is replicated across multiple nodes of the system, ensuring that the data is available even if some nodes fail. However, not all nodes have the same equal authority which means that it is not fully decentralized.

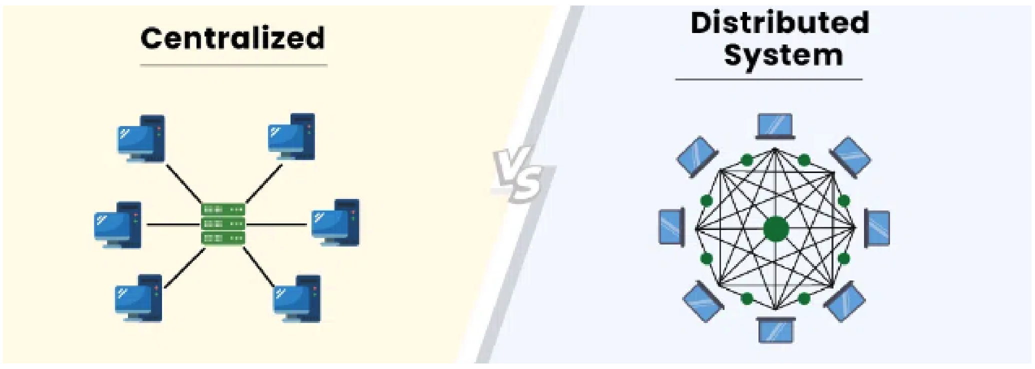

Figure 2.2: Centralized vs Distributed Systems [14].

In the context of education, distributed systems can significantly enhance the reliability and security of academic certificate validation. Educational institutions, accrediting bodies, and employers can benefit from a system where academic records are distributed across a network of trusted nodes, rather than being stored in a single central repository. The key benefits of this approach in the education sector include improved security, enhanced reliability and scalability.

2.4 Blockchain

Recently, another solution that have being proposed is the adoption of blockchain technology for academic certificate registry. As we can see in Figure 2.4, also being a distributed system, this technology goes beyond by offering a fully decentralized solution. Blockchain offers a decentralized, secure and *tamper-proof ledger* where certificates can be stored and verified. Tamper-proof ledger is a system designed to maintain records where once information is added, it cannot be altered or deleted. This is achieved through a combination of cryptographic techniques and a distributed network of computers (nodes) that each hold a copy of the ledger. Every transaction or entry is verified by these nodes, and any attempt to change past records would require altering the data on a majority of these nodes simultaneously, which is virtually impossible. This ensures the integrity and authenticity of the stored certificates, making them highly resistant to fraud and manipulation.

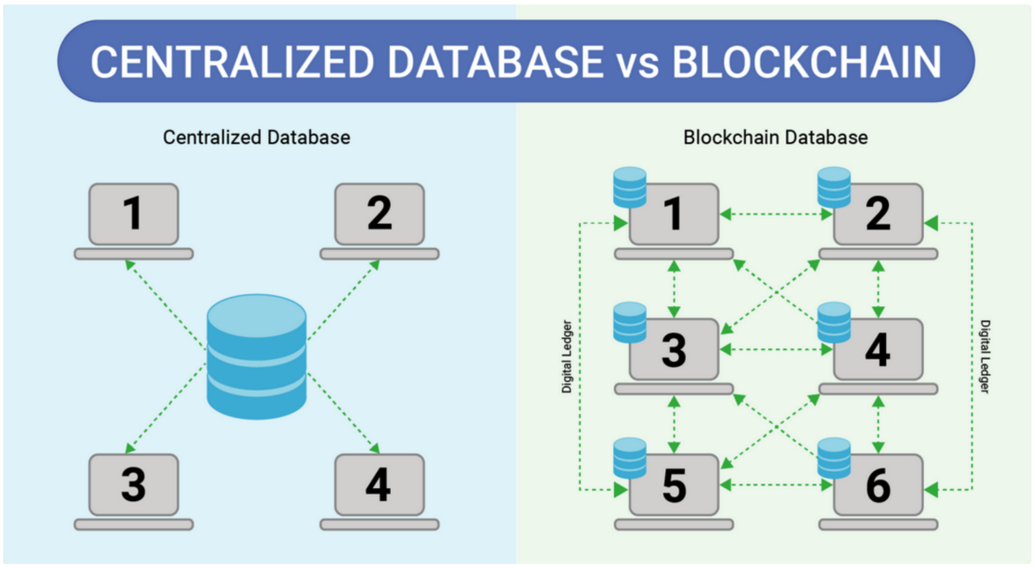

Figure 2.3: Centralized vs Blockchain Systems [6].

The use of blockchain technology ensures that certificates are immutable, transparent and accessible to all stakeholders [1]. Moreover, this technology enables the instant verification through cryptographic methods, eliminating the need for a central authority to manage the registry, thereby reducing the risk of fraud and manipulation.

In contrast to the traditional centralized data-based system, in our opinion, blockchain emerges with a huge influence capable of revolutionizing academic certificate registry systems. The decision to use blockchain technology as the foundation of our solution on the fact that blockchain technology is a key enabler of the *Web3* vision [3], which aims to create decentralized and fully distributed applications (*dApps*) that are secure, transparent and trust less where users have full control over their data and digital assets without having a **single point of failure**. For the implementation of a blockchain-based solution for our problem it is crucial to understand the foundational concepts that make this technology both revolutionary and reliable. Central to blockchain's efficacy is the principle of *distributed consensus* [37], which ensures the integrity, security and transparency of the ledger like mentioned before.

Blockchain technology started to gain popularity in 2008 initially described by Satoshi Nakamoto in a white paper entitled 'Bitcoin: A Peer-to-Peer Electronic Cash System' [22]. Although the term blockchain gained popularity in that year, with the introduction of Bitcoin cryptocurrency by Nakamoto, its underlying concepts have been used since the 1980s. Later in 2004, Harold Thomas Finney II introduced the Reusable Proof of Work (*RPOW*) system [11]. The RPOW system was a digital currency system that used a *proof-of-work* that limit the amount of work done by the server and to limit the amount of work done by the client. The RPOW system was the first system to use a blockchain-like structure to store and verify transactions. Later, in 2009 the first bitcoin transaction was made by Nakamoto to his friend Hal Finney [27] where was transferred 10 BTC (bitcoin). This marked the beginning of the blockchain technology era. In 2013, Vitalik Buterin proposed the concept of *smart contracts* in his white paper ‘Ethereum: The Ultimate Smart Contract and Decentralized Application Platform’ [4]. Upon this publication, *Ethereum* has launched his own blockchain in 2015. [28].

Blockchain can be defined as a time-ordered set of blocks or nodes where each block is cryptographically linked to the previous one forming a chain. All blocks are stored in a decentralized and distributed ledger and become trustworthy digital records what are unmodifiable in practice but very easy to verify. Like mentioned before, there is no centralized or hierarchical structure in the blockchain network and the information is shared by a network of *peers*.

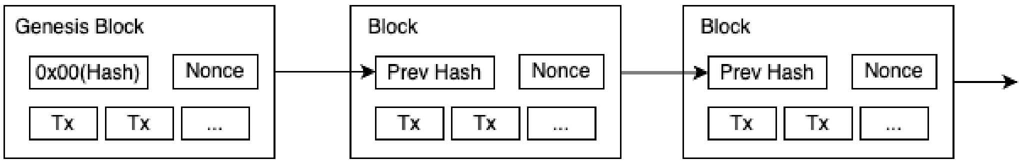

Figure 2.4: Blockchain structure adapted from [22]

As it can be observed in Figure 2.4, each block contains a reliable register of one or more actually executed transactions that are created and exchanged by the network participants (peers) which eventually must modify its state. To add new information to the chain, a **consensus** (2.4) about its truthfulness must be reached among the peers in the network.

The content of each transaction that is stored in a single block depends on the specific type of blockchain and its purpose. In our case and very succinctly, the transaction has an ‘item’ that contains information about the academic certificate; we will discuss this in more detail in the next chapters. Other example used nowadays is the Bitcoin, where the main information registered in transactions are exchanges of bitcoins between accounts.

Other important aspect of a chain node and the major reason for its security is the **hash** function. This function is used like a digital fingerprint to verify whether or not the data contained in the block has been tampered with. It is created when a new block is added or updated onto the chain. In the blockchain, each block’s hash includes the hash of the previous block, linking them together in a chain. If someone tries to change any information in a block, even just a tiny bit, the hash of that block will change completely. This change would break the link to the next block, making it obvious that the information has been tampered with. This is the reason why the blockchain is considered tamper-proof and secure.

Some blockchains support the use of *Smart Contracts* [19] which are self-executing contracts with the terms of the agreement directly written into code. These Smart Contracts are a critical component of several applications and platforms using a distributed ledger technology that we will be using in our solution and for better understanding we will explain with more detail in the next sections.

There are three main types of blockchain [26]:

- Public: is called public if each participants can read and use it to carry out transactions but also if everyone can participate in the process of creating the **consensus** which can be *Proof-of-Work* or *Proof-of-Stake*. In this type of blockchain there is no central authority nor a trusted third party to control the network. Examples of this type of blockchain are **Bitcoin** [22] and **Ethereum** [35]. The main advantages of this type of blockchain are:
  - High security and privacy,
  - Open and Flexible Environment,
  - No regulations,
  - Full Transparency and Systems,
  - Distributed, etc.
- Private: these are restricted and not open, such kind of blockchain also has features of access. This type of blockchains works mostly on closed systems and networks and are usually useful in organizations and companies which only selected members can join and access the data. Private blockchains have running

only authorized nodes and that means that no one from the outside of the private network is able to access the information and data exchanged between two nodes. In this type there is no mining, no proof of work, and no remuneration [16]. Examples of this type of blockchain are **Hyperledger Fabric** [12] and **R3 Corda** [10]. The main advantages of this type of blockchain are:

  - Full of privacy,
  - High Efficiency,
  - Faster Transaction,
  - Better Scalability,

- Consortium [18]: a combination of both public and private blockchains. As in a private blockchain, participants may join the network only by invitation and must be approved by the network owner, however, there is not a single organization that has control over the network. Instead, the control is distributed among a group of participants.
  - High Security,
  - High Scalability,
  - High Efficiency,
  - High Privacy,
  - High Flexibility,

Distributed Consensus Mechanisms

As mentioned in the beginning of the section 2.4, the blockchain technology is based on the concept of distributed consensus, which is a procedure used to achieve an agreement among all the peers of the blockchain network about the present state of the ledger. Through this mechanism, consensus algorithms ensure that all nodes in the network agree on the validity of the transactions and the order in which they are added to the blockchain. To do a parallelism with the real world, the consensus mechanism is the way that humans agree on the rules of a game, for example, in Monopoly where there are a lot of different ways to win, buying all the properties or end up with a lot of money in the bank and bankrupt all the other players but no matter what the rules are, everyone has agreed that it is a fair way to end the game [2]. Consensus mechanisms are essential to the security and integrity of the blockchain network, as they prevent malicious actors from altering the ledger and ensure that all transactions are valid and consistent across all nodes. It prevent the *double-spending* problem, where a user spends the same digital currency more than once, which is a common issue in digital currency systems.

- **Proof-of-Work (PoW)**: is the first consensus algorithm used in blockchain technology, introduced by Nakamoto in the Bitcoin white paper [22]. In this mechanism, miners compete to solve complex mathematical puzzles to validate transactions and add new blocks to the blockchain. The first miner to solve the puzzle is rewarded with a certain amount of cryptocurrency. PoW is known for its high energy consumption and slow transaction speeds, as miners must perform a large number of computations to solve the puzzle. However, it is also highly secure and resistant to attacks, as it would require a majority of the network's computing power to alter the blockchain. Examples of blockchains that use PoW are Bitcoin and Litecoin [33].
- **Proof-of-Stake (PoS)**: is an alternative consensus algorithm to PoW that aims to reduce energy consumption and increase transaction speeds. In PoS, validators are chosen to create new blocks based on the number of coins they hold, rather than the amount of computational power they contribute. Unlike PoW, PoS contributors (validators) are not rewarded with new coins, but with transaction fees. PoS is considered more energy-efficient than PoW, and more secure against 51% attacks, as it would require a majority of the network's coins to alter the blockchain. However, as the system is based on the number of coins held by the validators, it can lead to centralization, as those with more coins have more influence over the network.

After approaching all the three concepts of centralized, distributed and blockchain systems, we can conclude that the blockchain technology is the most suitable solution for the academic certificate registry system. This is due to its decentralized, secure and tamper-proof nature, which ensures the integrity and authenticity of academic certificates, while also providing instant verification and transparency to all stakeholders.

	Centralized Systems	**Distributed Systems**	**Blockchain**
Data Control	Centralized	Distributed	Fully Distributed
Data Storage	Single	Multiple	Multiple
Security	Low	Medium	High
Privacy	Low	Medium	High
Trustless	No	No	Yes
Scalability	Low	Medium	High

Table 2.1: Centralized vs Distributed vs Fully Distributed Systems

Chapter 3

Requirements

To meet the objectives of the SCAR project, the application to be developed, named DiGo Certify, should allow different stakeholders to access, share, and issue certificates on the Ethereum blockchain from anywhere in the world, offering a futuristic solution to the problems described in section (...).

The DiGo Certify application aims to address the needs of various stakeholders in the educational ecosystem. Mandatory and optional requirements have been defined based on essential functionalities and additional enhancements desired for the application.

.

3.1 Stakeholders

The SCAR solution involves three main stakeholders: students or alumni, employers or third-party entities, and college administrators. Each plays a crucial role in the application's ecosystem.

Role of Each Stakeholder and Actions They Can Perform:

- **Students or Alumni:** Students or alumni are the essence of our application. They will have the ability to register on the mobile application, where they can securely upload and store their academic certificates on the blockchain. Additionally, students or alumni can view and share their certificates with employers or third-party entities during recruitment processes, providing a transparent and efficient experience.
- **College Administrators:** College administrators are responsible for issuing and validating academic certificates provided by the educational institution. They will have access to the application to issue and authenticate certificates for students or alumni, thereby ensuring the integrity and validity of the documents.
- **Employers or External Entities:** Employers play a crucial role in requesting the verification of academic certificates during job interviews. By accessing the DiGo Certify platform, employers or external entities can instantly verify the authenticity of the certificates presented by students or alumni, without the need of possessing a wallet and with this ensuring a more reliable and transparent recruitment process.

3.2 Mandatory Requirements:

- **Implementation of the mobile application using React Native with Expo platform:** The choice of React Native along with the Expo platform provides an effective approach for developing multiplatform mobile applications. This ensures a consistent experience for users, regardless of the device used, and facilitates the development process for the development team.
- **Utilization of smart contracts in Solidity to store and validate academic certificates on the Ethereum blockchain:** The use of smart contracts in Solidity offers a secure and reliable solution for

storing and validating academic certificates on the Ethereum blockchain. This approach ensures data integrity and immutability, making the DiGo Certify platform highly reliable and resistant to fraud.

- **Development of features for registration, authentication, and secure storage of certificates on the blockchain:** Developing robust features for registering, authenticating, and securely storing certificates on the blockchain is essential to ensure the security and reliability of the DiGo Certify platform. These features should be designed with a focus on usability and security, providing an intuitive and transparent experience for users.

3.2.1 Optional Requirements:

- **Integration of additional features, such as real-time notifications and sharing of certificates through digital channels:** Incorporating extra functionalities like real-time notifications and certificate sharing through digital channels, such as social media platforms, can further enhance the user experience on the DiGo Certify platform. This enhancement can boost usability and attract new users to the platform.

3.3 Use Cases

The following use cases describe the interactions between the different stakeholders in the DiGo Certify platform. Each use case outlines the actions performed by the stakeholders and the expected outcomes of these interactions. The use cases are illustrated using diagrams based on the 4+1 View Model, which provides a comprehensive view of the system from multiple perspectives[20].

3.3.1 Use Case 1: Student Registration and Certificate Requesting

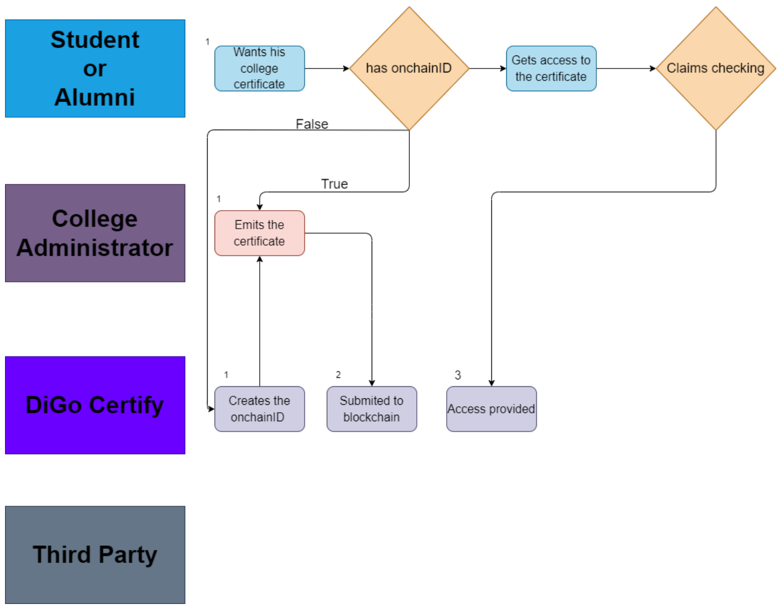

Figure 3.1: Use Case 1: Student Registration and Certificate Requesting.

- **Actors:** Student, College Administrator
- **Description:** A student registers on the mobile application and requests academic certificates from the college administrator.
- **Preconditions:** The student has access to the DiGo Certify platform. The college administrator has access to the platform and the student's academic records.
- **Postconditions:** The student receives the academic certificates and stores them on the blockchain.

- **Main Flow:**
- 1. The student downloads the `DiGo Certify` mobile application from the app store.
- 2. The student registers himself using their email address and connecting a wallet to the application.
- 3. The student requests academic certificates from the college administrator.
- 4. The college administrator issues the certificates, submitting them to the blockchain and emits the claims that will allow the student to access the certificates.
- 5. The student receives a notification that the certificates are available for download.
- **Alternative Flow:**
- 1.1 The student already has an account on the DiGo Certify platform.
- 1.2 The student logs in to their account using their email address.
- 1.3 The student requests additional academic certificates from the college administrator.
- 1.4 The college administrator issues the new certificates and submits them to the blockchain.
- 1.5 The student receives a notification that the certificates are available for download.
- **Exceptions:**
- 1. The student enters an invalid email address during registration.
- 2. The student fails to register on the application due to an error in the process.
- 3. The college administrator does not issue the certificates to the student.
- **Notes:** This use case illustrates the process of student registration and certificate requesting on the mobile application. It emphasizes the importance of secure and transparent interactions between the student and the college administrator, enabled by the blockchain technology.

3.3.2 Use Case 2: Certificate Validation by an External Entity

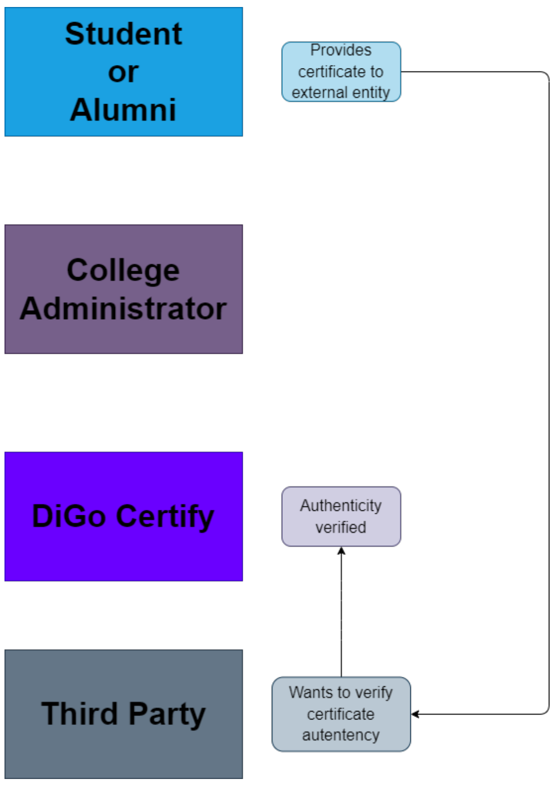

Figure 3.2: Use Case 2: Certificate Verification by an External Entity.

- **Actors:** External Entity, Student
- **Description:** For the purpose of a job application, an external entity verifies the authenticity of the student's academic certificates on the mobile application.
- **Preconditions:** The employer has access to the `DiGo Certify` application. The student has shared their certificates with the employer.
- **Postconditions:** The employer confirms the authenticity of the student's certificates and proceeds with the recruitment process.
- **Main Flow:**
- 1. The employer does not need to have a wallet to access the application.
- 2. The employer scans the QR code or accesses the link shared by the student to view the certificates.
- 3. The employer views the student's certificates and verifies their authenticity on the blockchain.
- 4. The employer confirms the authenticity of the certificates and proceeds with the recruitment process.
- **Exceptions:**
- 1. The employer fails to verify the authenticity of the certificates due to an error in the process.
- 2. The employer does not confirm the authenticity of the certificates.
- **Notes:** This use case illustrates the process of certificate verification by an employer on the `DiGo Certify` mobile application. It emphasizes the importance of transparency and trust in the recruitment process, enabled by the blockchain technology.

3.3.3 Use Case 3: Student or Alumni requests a change to the certificate

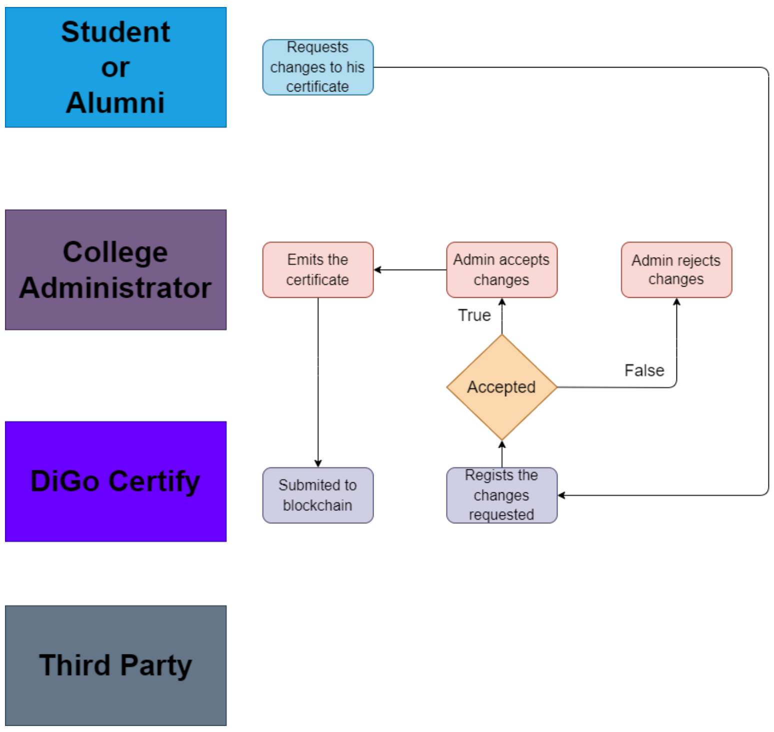

Figure 3.3: Use Case 3: Student or Alumni requests a change to the certificate.

- **Actors:** Student, College Administrator
- **Description:** A student or alumni requests a change to the academic certificates issued by the college administrator.
- **Preconditions:** The student has access to the DiGo Certify platform. The college administrator has access to the platform and the student's academic records.
- **Postconditions:** The student receives the updated academic certificates and stores them on the blockchain.
- **Main Flow:**
- 1. The student logs in to the `DiGo Certify` mobile application.
- 2. The student requests a change to the academic certificates from the college administrator.
- 3. The college administrator updates the certificates, submitting them to the blockchain and emits the claims that will allow the student to access the updated certificates.
- 4. The student receives a notification that the updated certificates are available for download.
- **Exceptions:**
- 1. The student fails to log in to the application due to an error in the process.
- 2. The student does not request a change to the academic certificates.
- 3. The college administrator does not update the certificates for the student.
- **Notes:** This use case illustrates the process of requesting a change to academic certificates on the mobile application. It emphasizes the importance of maintaining accurate and up-to-date records on the blockchain, enabled by the blockchain technology.

Chapter 4

Solution Architecture

The present chapter covers the system's components, their interactions, and the underlying technologies used to implement the solution. The architecture is designed to ensure data integrity, security, and scalability while providing a seamless user experience. We will cover the concept of a multiplatform application, how it functions, the various solutions available, and a detailed discussion of the chosen technology stack, specifically React Native with the Expo[5] platform.

4.1 Architecture Overview (CHAPTER UNDER DEVELOPMENT)

The architecture of the `DiGo Certify` system is designed to be modular, scalable, and secure. The system 4.1 is divided into two main layers: the mobile application layer and the fully distributed layer. Each layer has specific responsibilities and interacts with the others to deliver the desired functionality.

Mobile Application Layer

The mobile application layer consists of the DiGo Certify application, which is built using React Native and Expo. This application is responsible for the user interface and handles interactions with the end-users. The choice of React Native ensures that the application is cross-platform, providing a seamless experience on both iOS and Android devices. The application allows users to register, authenticate, and manage their certificates. It communicates with the distributed layer to store and validate these certificates on the Ethereum blockchain.

Fully Distributed Layer

The fully distributed layer is implemented using the Ethereum blockchain. It consists of smart contracts written in Solidity, which handle the storage and validation of academic certificates. This layer leverages the security and immutability of the blockchain to ensure the integrity of the certificates. The interaction between the mobile application and the blockchain is facilitated by a provider, such as Hardhat, which helps in deploying and managing the smart contracts.

Below is a high-level diagram of the DiGo Certify architecture:

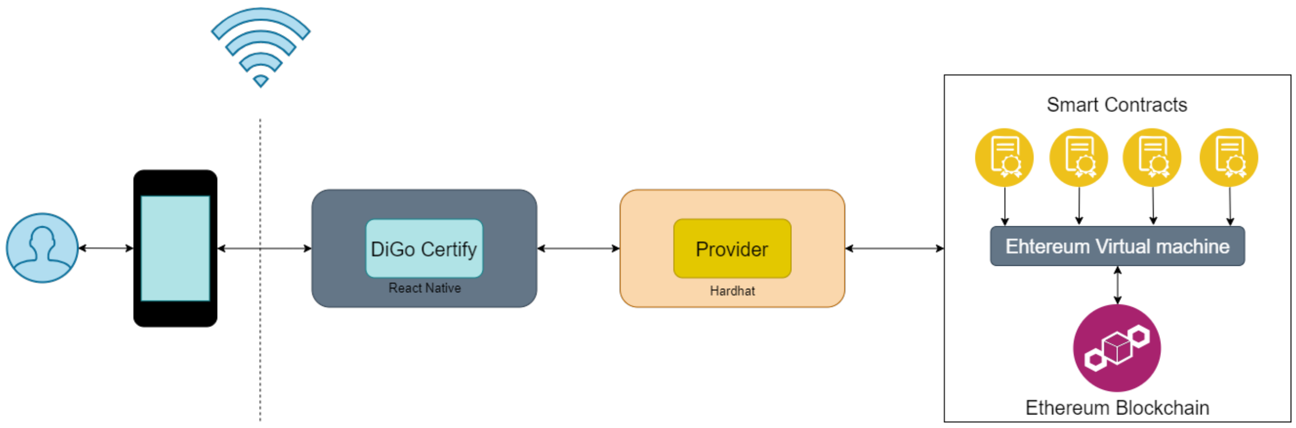

Figure 4.1: Architecture Overview Diagram (inspired by [13]).

4.2 Fully Distributed Environment

The fully distributed environment of the DiGo Certify system leverages blockchain technology to create a secure, decentralized, and transparent platform for managing academic certificates. By using the Ethereum blockchain, the system ensures that certificates are immutable and verifiable by any third party, thereby eliminating the risk of fraud and enhancing trust in the certification process.

4.2.1 Smart Contracts on Ethereum

At the core of the distributed environment are the smart contracts deployed on the Ethereum blockchain. Smart contracts are self-executing contracts with the terms of the agreement directly written into code. In the context of DiGo Certify, these smart contracts handle the issuance, validation, and storage of academic certificates. Each certificate is represented as a unique digital asset on the blockchain, and its authenticity can be easily verified by checking the blockchain records.

The smart contracts are written in Solidity, a statically-typed programming language designed for developing smart contracts that run on the Ethereum Virtual Machine (EVM). The key functions of these smart contracts include:

- **Certificate Issuance:** This function allows authorized entities, such as educational institutions, to issue certificates to students. The institution submits a transaction to the smart contract, including details such as the student's name, course, and date of completion. Upon verification, the smart contract records the certificate on the blockchain.
- **Certificate Validation:** This function enables third parties, such as employers, to verify the authenticity of a certificate. By querying the blockchain, they can confirm that the certificate was indeed issued by a legitimate institution and has not been altered.
- **Certificate Storage:** Certificates are stored in a decentralized manner on the blockchain. This ensures that they are *tamper-proof* (i.e., they cannot be altered or manipulated) and can be accessed by anyone with the appropriate permissions.

4.2.2 Ethereum Virtual Machine (EVM)

The Ethereum Virtual Machine (EVM) is a decentralized computing environment that executes smart contracts on the Ethereum network. The EVM ensures that the execution of smart contracts is consistent and secure across all nodes in the network[23]. This decentralized nature of the EVM makes it highly resistant to hacking and fraud, as any attempt to alter a certificate would require the attacker to gain control of a majority of the network's computing power, which is practically impossible. Once a certificate is recorded on the blockchain, it cannot be altered or deleted, ensuring that the certificates remain trustworthy and verifiable indefinitely.

Additionally, all transactions on the Ethereum blockchain are publicly accessible, providing a transparent record of certificate issuance and validation, which enhances trust in the certification process.

4.2.3 Provider (Hardhat)

Hardhat is a comprehensive development environment for Ethereum software, providing an array of tools for compiling contracts, running tests, and deploying them to the blockchain. It significantly simplifies the development and deployment process, making it easier to manage the smart contracts that underpin the DiGo Certify system.

During the development phase, Hardhat is used to compile Solidity code into bytecode that can be executed by the Ethereum Virtual Machine (EVM). It allows developers to write and run tests, ensuring that smart contracts behave as expected. Hardhat also provides scripts for deploying smart contracts to the Ethereum network, making the deployment process straightforward and repeatable.

In addition to its development capabilities, Hardhat also plays a role in facilitating the interaction between the mobile application and the blockchain during the execution phase. This ensures smooth and secure communication between the components of the DiGo Certify system. Although primarily a development tool, Hardhat's robust features and integration capabilities make it useful throughout the lifecycle of the smart contracts.

In the context of the DiGo Certify system, Hardhat serves as a development and deployment environment, while also supporting interaction with the blockchain during execution. It's important to note that other tools and services, such as Infura or Alchemy, can be used as blockchain providers during execution. These services offer additional capabilities, such as scalable node infrastructure and advanced monitoring, but in our implementation, Hardhat has proven sufficient for both development and interaction purposes.

4.2.4 Security

Security and scalability are critical considerations in the design of the DiGo Certify system. The Ethereum blockchain's inherent security features, such as its decentralized consensus mechanism and cryptographic algorithms, providing a robust foundation for secure certificate management. Additionally, the use of smart contracts ensures that certificate issuance and validation processes are automated and tamper-proof[36].

The fully distributed environment of DiGo Certify leverages the strengths of blockchain technology to provide a secure, transparent, and scalable platform for managing academic certificates.

4.3 Mobile Application

4.3.1 Multiplatform Application

A multiplatform application is designed to run seamlessly on multiple operating systems, such as iOS and Android, using a single codebase. This approach significantly reduces development time and costs while ensuring a consistent user experience across different devices. The core idea is to write the code once and deploy it across multiple platforms, which is particularly beneficial for applications that need to reach a broad audience.

Functionality and Operation

Multiplatform applications leverage frameworks that provide tools and libraries to facilitate cross-platform development. These frameworks abstract away the differences between the various platforms, allowing developers to focus on building features rather than dealing with platform-specific nuances. The primary goal is to achieve native-like performance and look-and-feel while maintaining a shared codebase.

Available Solutions

Several frameworks are available for developing multiplatform applications, each with its own set of features and trade-offs. The most notable ones include:

- React Native[8]
- Flutter[15]
- Kotlin Multiplatform Mobile (KMP)[17]

- React Native[8]
- Flutter[15]
- Kotlin Multiplatform Mobile (KMP)[17]

React Native

React Native is a popular open-source framework developed by Facebook. It allows developers to build mobile applications using JavaScript and React, a widely-used library for building user interfaces. React Native bridges the gap between web and mobile development by enabling code reuse across platforms while providing near-native performance[8].

Key Features:

- **Component-Based Architecture:** Enables modular and maintainable code.
- **Hot Reloading:** Allows developers to see changes in real-time without recompiling the entire application.
- **Rich Ecosystem:** A vast collection of libraries and tools that streamline development.
- **Community Support:** Extensive community contributions and support.

Flutter

Flutter, developed by Google, is another powerful framework for building natively compiled applications for mobile, web, and desktop from a single codebase. It uses the Dart programming language and provides a rich set of pre-designed widgets to create highly customizable interfaces[15].

Key Features:

- **Hot Reload:** Similar to React Native's hot reloading, enabling quick iterations.
- **Expressive UIs:** Rich set of customizable widgets.
- **Performance:** Compiled directly to native code, which can lead to better performance.

Kotlin Multiplatform Mobile (KMP)

KMP, developed by JetBrains, allows developers to use Kotlin for developing iOS and Android applications. It focuses on sharing code, particularly business logic, while allowing platform-specific code where necessary[17].

Key Features:

- **Code Sharing:** Share common code across platforms while writing platform-specific code when needed.
- **Native Performance:** Utilizes native components and performance optimizations.

Table 4.1: Comparison of Flutter, React Native, and Kotlin.[32]

Parameter	**Flutter**	**React Native**	**Kotlin**
Ease of Coding	Widget-based, reactive framework (Dart)	JSX and JavaScript, component-based (React)	Concise, expressive, reduces boilerplate (Kotlin)
Scalability	Good scalability, reactive architecture	Scales well, may need native modules	Excellent scalability, especially for large codebases
Performance	High performance, compiled language (Dart)	Near-native performance, native modules if needed	Efficient performance, native language for Android
Learning Curve	Moderate, widget-based approach	Relatively easy for JavaScript developers	Moderate, especially for those familiar with Java
Popularity	Growing rapidly, gaining popularity	Widely adopted, mature, and well-established	Highly popular for Android development
Stability	Stable, regular updates and improvements	Stable, backed by Facebook and active community	Stable, regular updates and improvements
Types of App Dev	Cross-platform focus for consistent UI	Cross-platform, leverages JavaScript expertise	Primarily for native Android; Kotlin Multi Platform for cross-platform
Community Support	Growing community support	Large and active community support	Strong support, especially in the Android community

4.3.2 Chosen Solution: React Native with Expo

For the `DiGO Certify` application, we chose React Native with the `Expo` platform. This decision was influenced by several factors, including our team's familiarity with JavaScript and React, the maturity and stability of the React Native ecosystem, and the added benefits provided by Expo[5].

To make an informed decision, we compared several frameworks, including Flutter and Kotlin, based on various parameters 4.3.2.

React Native's component-based architecture aligns well with our need for a modular and maintainable codebase. Our team's existing knowledge of JavaScript and React significantly reduced the learning curve, allowing us to quickly become productive and focus on delivering features. Compared to Flutter and Kotlin, React Native's learning curve is relatively easy for JavaScript developers, while Flutter requires learning Dart, and Kotlin, despite being expressive and reducing boilerplate, may be more moderate for developers familiar with Java.

In terms of performance, React Native provides near-native performance, ensuring that our application runs smoothly on both iOS and Android[9]. This is similar to Flutter 4.3.2, which offers high performance through its compiled language, Dart, and Kotlin, which provides efficient performance for native Android development. However, React Native can scale well with the potential need for native modules, providing flexibility in development.

The framework's rich ecosystem of libraries and tools further accelerated our development process, providing pre-built components and solutions that we could easily integrate into our application. The extensive community support for React Native ensured that we had access to numerous resources, tutorials, and third-party libraries, which proved invaluable during the development process. This support network allowed us to quickly troubleshoot issues and implement best practices, contributing to a more efficient development cycle. While Flutter's community is rapidly growing, React Native's community is already large, mature, and well-established, and Kotlin also enjoys strong support, especially in the Android community.

Expo enhances React Native by offering a suite of tools and services that simplify development[5]. With Expo, we benefit from an integrated environment for developing, building, and deploying React Native applications. The platform's managed workflow handles many of the complexities of building and deploying mobile applications, allowing us to focus on developing features rather than dealing with infrastructure. Expo's easy setup and configuration process streamlined our project initialization, while its over-the-air update capability enables us to push updates to users without requiring a full app store review process.

The development workflow for DiGo Certify using React Native and Expo involves several key steps. Initially, we set up the project with Expo CLI, which provides a streamlined setup process and essential tools. We then focused on building the application UI using React Native's component-based approach. This method allows us to create reusable UI elements that help maintain consistency and simplify development. The development workflow for DiGo Certify using React Native and Expo involves several key steps. Initially, we set up the project with Expo CLI, which provides a streamlined setup process and essential tools. We then focused on building the application UI using React Native's component-based approach. This method allows us to create reusable UI elements that help maintain consistency and simplify development.

For integrating blockchain functionality, we implemented smart contracts in Solidity and connected them with the React Native application through ethers.js. This integration enables secure interactions with the Ethereum blockchain, allowing for the storage and validation of academic certificates. For integrating blockchain functionality, we implemented smart contracts in Solidity and connected them with the React Native application through ethers.js. This integration enables secure interactions with the Ethereum blockchain, allowing for the storage and validation of academic certificates.

Testing and debugging are facilitated by Expo's built-in tools, which allow us to test the application on various devices and simulators. This ensures that our application performs well across different platforms and devices. Finally, Expo simplifies the deployment process with its build and publish services, allowing us to distribute the application through app stores seamlessly.

In summary, the choice of React Native with Expo for DiGo Certify provides a robust, efficient, and scalable solution for developing a secure and user-friendly multiplatform application. This architecture leverages modern technologies to meet the needs of our diverse user base, ensuring a high-quality user experience across all supported devices. In summary, the choice of React Native with Expo for DiGo Certify provides a robust, efficient, and scalable solution for developing a secure and user-friendly multiplatform application. This architecture leverages modern technologies to meet the needs of our diverse user base, ensuring a high-quality user experience across all supported devices.

Chapter 5

Implementation

This is where the implementation goes to.

Chapter 6

Work Plan

This chapter describes the work plan for the project.

Bibliography

[1] A. Alammary, S. Alhazmi, M. Almasri, and S. Gillani. Blockchain-based applications in education: A systematic review. *Applied Sciences*, 9(12):2400, 2019.

[2] B. Becher and B. Whitfield. What is a consensus mechanism? Available at: https://builtin.com/blockchain/consensus-mechanism, 2023.

[3] A. Buldas, D. Draheim, M. Gault, and M. Saarepera. Towards a foundation of web3. In *International Conference on Future Data and Security Engineering*, pages 3–18. Springer, 2022.

[4] V. Buterin. Ethereum: the ultimate smart contract and decentralized application platform. *Libro blanco de Ethereum. Available at: http://web.archive.org/web/20131228111141/http://vbuterin.com/ethereum.html*, 8, 2013.

[5] C. Cheever. Expo technologies inc. Available at: https://expo.dev/, 2015.

[6] S. DB. Blockchain database. Available at: https://www.scylladb.com/glossary/blockchain-database/.

[7] C. Delgado-von Eitzen, L. Anido-Rifón, and M. J. Fernández-Iglesias. Nfts for the issuance and validation of academic information that complies with the gdpr. *Applied Sciences*, 14(2), 2024.

[8] B. Eisenman. *Learning React Native: Building Native Mobile Apps With Javascript 2nd Edition*. Oreilly & Associates Inc, 2017.

[9] Facebook. React native performance. Available at: https://facebook.github.io/react-native/docs/performance, 2024.

[10] M. Farina. Corda distributed ledger platform. Available at: https://r3.com/, 2017.

[11] H. Finney. Rpow-reusable proofs of work. Available at: https://nakamotoinstitute.org/finney/rpow/index.html, 2004.

[12] H. Foundation. Hyper ledger fabric. Available at: https://www.hyperledger.org/projects/fabric, 2016.

[13] GeeksforGeeks. What are dapps (decentralized applications)? Available at: https://www.geeksforgeeks.org/architecture-of-a-dapp/, 2022.

[14] GeeksforGeeks. Centralized vs distributed system. Available at: https://www.geeksforgeeks.org/centralized-vs-distributed-system/, 2024.

[15] Google. Flutter. Available at: https://flutter.dev/, 2018.

[16] D. Guegan. Public Blockchain versus Private blockhain, Apr. 2017. Documents de travail du Centre d'Economie de la Sorbonne 2017.20 - ISSN : 1955-611X.

[17] JetBrains. Kotlin multiplatform. Available at: https://kotlinlang.org/, 2011.

[18] N. R. Kasi, R. S, and M. Karuppiah. Chapter 1 - blockchain architecture, taxonomy, challenges, and applications. In S. H. Islam, A. K. Pal, D. Samanta, and S. Bhattacharyya, editors, *Blockchain Technology for Emerging Applications*, Hybrid Computational Intelligence for Pattern Analysis, pages 1–31. Academic Press, 2022.

[19] G. Kaur, A. Habibi Lashkari, I. Sharafaldin, and Z. Habibi Lashkari. Introduction to smart contracts and defi. In *Understanding Cybersecurity Management in Decentralized Finance: Challenges, Strategies, and Trends*, pages 29–56. Springer, 2023.

[20] P. Kruchten. The 4+1 view model of software architecture. Available at: `https://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf`, 1995.

[21] Lusa. Ordem apresentou queixa contra advogado suspeito de falsificar certificados académicos. Available at: `https://www.publico.pt/2022/06/20/sociedade/noticia/ordem-apresentou-queixa-advogado-suspeito-falsificar-certificados-academicos-2010734`, 2022.

[22] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Available at SSRN: https://ssrn.com/abstract=3440802 or http://dx.doi.org/10.2139/ssrn.3440802*, August 2008.

[23] O. Network. What is ethereum virtual machine (evm) in blockchain? Available at: `https://medium.com/@orderlynetwork/what-is-ethereum-virtual-machine-evm-in-blockchain-e7db57167bea`, Year of Publication. Accessed: 02/06/2024.

[24] J. E. Olson. Chapter 5 - origins of a database archiving application. In J. E. Olson, editor, *Database Archiving*, The MK/OMG Press, pages 71–84. Morgan Kaufmann, Boston, 2009.

[25] K. Omai. Rid country of fake academic papers menace. Available at: `https://nation.africa/kenya/blogs-opinion/blogs/rid-country-of-fake-academic-papers-menace-4598654`, 2024.

[26] P. Paul, P. Aithal, R. Saavedra, and S. Ghosh. Blockchain technology and its types—a short review. *International Journal of Applied Science and Engineering (IJASE)*, 9(2):189–200, 2021.

[27] A. Peterson. Hal finney received the first bitcoin transaction. here's how he describes it. *Washington Post, available at: https://www.washingtonpost.com/news/the-switch/wp/2014/01/03/hal-finney-received-the-first-bitcoin-transaction-heres-how-he-describes-it/*, 3, 2014.

[28] N. Reiff. Bitcoin vs. ethereum: what's the difference. *Investopedia, available at: `https://www.investopedia.com/articles/investing/031416/bitcoin-vs-ethereum-driven-different-purposes.asp`*, 2020.

[29] O. S. Saleh, O. Ghazali, and M. E. Rana. Blockchain based framework for educational certificates verification. *Journal of critical reviews*, 2020.

[30] A. Saxena. Types of databases. Available at: `https://www.scaler.com/topics/dbms/types-of-database/`, 2023.

[31] M. Sharples and J. Domingue. The blockchain and kudos: A distributed system for educational record, reputation and reward. In *Adaptive and Adaptable Learning: 11th European Conference on Technology Enhanced Learning, EC-TEL 2016, Lyon, France, September 13-16, 2016, Proceedings 11*, pages 490–496. Springer, 2016.

[32] A. Smith. Flutter vs react native vs kotlin multiplatform. Available at: `https://www.hiddenbrains.com/blog/flutter-vs-react-native-vs-kotlin.html/`, 2024.

[33] I. Takashima. Litecoin: The ultimate guide to the world of litecoin, litecoin crypocurrency, litecoin investing, litecoin mining, litecoin guide, cryptocurrency. *CreateSpace Independent Publishing Platform*, 2018.

[34] B. Thomas. Linways technologies pvt. ltd. Available: `https://blog.linways.com/why-linways-academic-management-system/`, 2013.

[35] S. Tual. Ethereum launches. *Ethereum Foundation Blog*, 30, 2015.

[36] G. Wood. Ethereum: A secure decentralised generalised transaction ledger. Available at: `https://ethereum.org/en/whitepaper/`, 2014.

[37] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou. A survey of distributed consensus protocols for blockchain networks. *IEEE Communications Surveys & Tutorials*, 22(2):1432–1465, 2020.