2019

A Blockchain approach for detecting counterfeit academic certificates in Kenya

Joy A. Otuya
Faculty of Information Technology (FIT)
Strathmore University

A Blockchain Approach for Detecting Counterfeit Academic Certificates in Kenya

Otuya Joy Atuwo

Master of Science in Information Technology

2019

A Blockchain Approach for Detecting Counterfeit Academic Certificates in Kenya

Otuya Joy Atuwo

096422

Submitted in Partial Fulfilment of the Requirements for the Degree of Master of Science in Information Technology at Strathmore University

Faculty of Information Technology
Strathmore University
Nairobi, Kenya

June, 2019

DECLARATION

I declare that this work has not been previously submitted and approved for the award of a degree by this or any other University. To the best of my knowledge and belief, the proposal contains no material previously published or written by another person except where due reference is made in the proposal itself.

Otuya Joy Atuwo

Signature…….....

Date…….....

Approval

The thesis of Otuya Joy Atuwo was reviewed and approved by the following:

Dr. Vitalis Ozianyi (PhD),

Academic Director, Faculty of Information and Technology,

Strathmore University

Dr. Joseph Orero (PhD),

Dean, Faculty of Information and Technology,

Strathmore University

Professor Ruth Kiraka (PhD),

Dean, School of Graduate Studies,

Strathmore University

ABSTRACT

There has been an increase in the number of people seeking to pursue higher education and secure employment in different sectors across Kenya and the world at large. Consequently, there has been an exponential increase in the number of fake academic certificates being issued. Employers end up higher less qualified staff, while education institutions admit students with lesser qualifications. This has had a negative impact on education institutions in as far as credibility is concerned and for the employers, low productivity from the under qualified staff.

Several methods have been put in place by different organizations for purposes of verifying academic certificates in Kenya. However, most of the methods employed are manual processes which are time consuming, tiresome and more prone to errors. The automated processes currently in use still have loopholes that can easily be exploited to bypass the verification process. For this reason, there is need to come up with an automated solution that will ensure proper verification of academic certificates thereby upholding the credibility of our academic institutions and increasing productivity in other organizations.

This research explores different strategies employed in Kenya together with their challenges, the traditional and automated systems available for verification of academic certificates and finally proposed the development of a blockchain capable application for purposes of verifying academic certificates.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND ACRONYMS

ACL - Access Control List

CA - Certificate Authority

CLI - Command Line Interface

DpoS - Delegated Proof of State

ECDSA - Elliptic Curve Digital Signature Algorithm

IBM - International Business Machines

IoT - Internet of Things

KASNEB - Kenya Accountants and Secretaries National Examinations Board

KCSE - Kenya Certificate of Secondary Education

KENET - Kenya Education Network

KNQA - Kenya National Qualifications Authority

NoSQL - No Structured Query Language

PoW - Proof of Work

PoS - Proof of State

QR - Quick Response

RAM - Random Access Memory

RFID - Radio Frequency Identifier

SHA - Secure Hash Algorithm

DEFINITION OF TERMS

Asset – Anything that can be owned or controlled to produce value. In this research an asset refers to an academic certificate (Apte & Petrovsky, 2016).

Participant – An entity that is involved in transaction process in a business network. In this study a participant refers to a business network/fabric administrator (Hyperledger, 2018).

Transaction – This refers to the process of initiating an update to the ledger. It may be transferring an asset from one participant to another, adding an asset or participant.

Business Network – Network that allow participants to exchange assets among each other through transactions (Hyperledger, 2018).

ACKNOWLEDGEMENTS

I give glory to God for enabling me to complete my Thesis.

Special gratitude goes to my supervisor Dr. Vitalis Ozianyi for the guidance and support accorded throughout starting from the proposal stage till the final thesis submission stage. I would also wish to acknowledge the Faculty of Information Technology at Strathmore for the tireless support and timely updates during my Msc. Information technology Period at Strathmore University.

My utmost gratitude also goes to my Family and friends who stood by me during my course work period as well as the Thesis writing stage.

Lastly, I wish to thank all the respondents who managed to spare their time to respond to my questions in the informal interviews conducted during this research period.

DEDICATION

I dedicate this work to my beloved husband James Oyim, my parents Janet and Prof. Robert Otuya, my siblings Grace and David Otuya and to all who have gave me ample time to successfully complete my project work.

CHAPTER ONE

INTRODUCTION

1.1 Background to the Study

Falsification of academic certificates has become an issue of global concern. (New York Times, 2015) pointed out that there existed more than 3300 diploma mills in United States of America. Other cases include alteration of genuine certificates to meet qualification that are being sought for by employers.

In Africa and more so Kenya the issue of academic certificate falsification poses a huge threat to the economy and job market. The fake certificates take the form of falsified documents, academic certificate impersonation and the existence of diploma mills that have been on the rise in Kenya (Solomon, 2016). South Africa also witnessed a higher increase in the use of fake academic certificates in the financial year 2017/2018 where more than 982 fraudulent qualifications were uncovered (Maqhina, 2018)

The increase in the use of falsified documents is majorly attributed to the valued laid on possession of undergraduate or post graduate degrees and diplomas for chances of securing employment or getting promoted. Many countries and organization do not have adequate mechanisms instituted to counter the menace of academic certificate falsification with a good number still relying on manual certificate verification methods. This has ended up damaging the reputation of many academic institution and locking out genuine candidates from the job market (Garwe, 2014).

It is therefore necessary for a more reliable system of certificate verification to be put in place to counter the fraudulent qualifications menace. The most appropriate will be to adopt the use of automated certificate verification methods such as the use of blockchain technology.

The blockchain technology is one of the revolutionary technologies since the invention of the Internet and is poised to change Information Technology just as open source software did. Many companies across the globe are engaging in research towards this new technology to see how they can incorporate it in their day to day activities.

A blockchain can be viewed as a distributed database of records, a public ledger consisting of all transactions made or simply digital events executed and shared among a group of people. Once information is entered into the database it can never be changed. This make blockchain to be incorruptible digital ledgers. Because of their incorruptible nature, blockchain technology has been applied across different sectors such as record keeping. There are private blockchains confined to internal organization matters and public blockchains where any can access and add information. Every block in the blockchain holds identical information that cannot be altered but only viewed by anyone in possession of the cryptographic public key (Sharples & Domingue, 2016). Due to their incorruptible nature, immutability and provenance, blockchain is seen a technology that could be used to safeguard academic awards such as certificates.

1.2 Problem Statement

There has been an increase in the number of forgeries of academic certificates in Kenya and around the globe. This is attributed to the absence of proper certificate verification mechanisms (Garwe, 2014) . Academic certificate falsification has consequently put the credibility of the Kenyan higher education system in question while damaging the reputation of innocent academic institutions whose certificates are issued without their knowledge and consent. On the other hand, Employers end up placing people into roles they do not qualify for, leading to serious financial and productivity loses for the organizations (Jimu, 2018).

The issues highlighted show that there exists great need for more efficient ways to be adopted for purposes of verifying academic certificates. The blockchain technology was used in this research for purposes of verifying academic certificates.

1.3 General Objective

The aim of this study was to develop a blockchain application to counter forgery of university certificates by publicly providing evidence that a student received a certificate from a certain university in Kenya.

1.3.1 Specific Objectives

i. To identify the challenges faced in academic certificate verification.
ii. To analyse the different techniques and models used for detecting academic certificate forgery.
iii. To review existing frameworks used for the development of the blockchain applications.
iv. To design, develop and test a blockchain application to counter forgery of university certificates in Kenya.

1.3.2 Research Questions

i. What challenges are currently faced in verification of academic certificates?
ii. What are some of the existing techniques and models used for detecting education certificate forgery?
iii. What are the existing frameworks for developing blockchain applications?
iv. How can a blockchain application of verifying university academic certificates be developed and tested?

1.4 Justification of the Study

A few mechanisms have been put in place to aid in verification of academic certificates. These include the use of both manual and automated methods. However, there is still great need of having a centralised efficient and reliable system of verifying academics certificates to ensure that the credibility of employees and academic institutions in Kenya is upheld. Blockchain technology is one technology that would come in handy to safeguard academic awards such as certificates since it offers great transparency, enhanced security as compared to other record-keeping systems, improved traceability, increased efficiency and speed as well as reducing business costs (Lemieux, 2016).

A further review of literature revealed that academic research on the use of blockchain technology for purposes of verifying university certificates in Kenya has been minimal. This additional justifies the need of conducting the research.

1.5 Scope and Limitation

The blockchain application will be limited to verification of university certificates in Kenya. Due to time and resource limitation the final product was a prototype covering the primary functionality of the system. User acceptance of the final product was also not part of the researcher's scope.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

The demand for a highly skilled and educated labour force and the desire to advance careers has led to an increase in the number of people seeking to pursue further education in Universities and colleges. This has consequently increased the use of fake academic certificates by candidates when looking for jobs or when seeking to pursue further studies. Several measures have been put in place by various private and public sector organizations, colleges and universities to alleviate academic document forgery. However, many cases still pass undetected (Some, 2017). This research shows the various measures currently in place to detect document forgery and the challenges faced in the whole process. The research also discusses the traditional and automated systems available to alleviate academic document forgery and finally the development and testing of a blockchain application for countering academic certificate forgery in Kenya.

2.2 Motivation of Acquiring or Issuing Falsified Academic Certificates

Academic document forgery can be traced back to the 11th century. Several cases of academic document forgery have been reported across different parts of the globe. Kenya's print and electronic media have cited several cases of fake certificates and this trend is on the rise. This is attributed to the global value of educational awards, specifically higher education certificates. Many institutions and individuals have taken advantage of gullible customers who are in such of acquiring academic qualifications to secure jobs and sell them fake certificates.

Four major processes of academic document falsification are described by Decoo, (2002). These include:

i. **Translations:** These involves deliberate alteration of documents with the aim of providing false information. Grades and course titles can be changed to suite the requesting party requirements.
ii. **In-house production:** Employees of a university or college are involved in alteration and issuance of falsified documents. The certificates are modified but will appear with

authentic seals, watermarks, stamps and legitimate signatures. In most cases, certificates are awarded for unfinished degree programs that a candidate never registered in.

iii. **Degree or Diploma mills:** These are institutions that pose as higher education institutions but sell illegitimate degrees and diplomas. In most cases, individuals who receive certificates from these organizations aren't aware that the awards obtained aren't legitimate.

iv. **Document fabrication:** These involves designing fake documents to appear as being issued by a legitimate university or college.

2.3 Existing Methods of Detecting Academic Document Falsification

2.3.1 Traditional Methods of Verifying Academic Certificates

Most organization still rely on manual methods of verifying certificates. This is usually a very cumbersome and time-consuming process. The universities can choose to do the actual verification by contacting the awarding institution of a certificate to request for a certified copy of the certificate. This can sometimes take several days to months depending on the response from the issuing authority. Third party verification is also another method used where an institution subcontracts the verification process to another company. The third-party companies charge huge amounts of money to verify documents. The general elements checked on the certificates include; the institution's seal, stamp, correct date and signature(Asadi, Rahbar, Rezvani, & Asadi, 2018).

Advancement in technology has made this verification process a challenge as counterfeiters produce perfect replicas of the legitimate documents. A few incidences that may raise eyebrows on the validity of an academic certificate include: evidence of corrected personal data, an applicant claims to have lost an original document or the names on the certificates do not match with original documents such as identity cards (Malkawi, 2017; Van Tol, 1990).

The Kenya National Qualifications Authority (KNQA) is an example of a third-party verification organisation that was setup by the Government of Kenya to verify and weed out fake certificates in the country. KNQA currently uses manual methods of verifying certificates through its VeriCert program. An applicant who wishes to have a certificate verified sends the certificates

copied and pays a fee of Ksh.1000 for the verification process. However, KNQA is in the process of developing a system that will consolidate all academic institution certificates into a central database where third party queries will be done. The database is set to help in verification of KCSE, KASNEB, TVET level and University Level Certificates (KNQA_ICT, 2018).

2.3.2 Automated Methods of Verifying Academic Certificates

Kaibiru and Shibwabo (2017) proposed a method for authenticating Kenya Certificate of Secondary Education Certificates. The proposed architecture made use of protected Quick Response (QR) codes and cryptographic techniques to verify certificates. The solution involved hashing certificate details using SHA, generating digital signatures, converting the digital signatures to QR codes and finally appending the QR code onto the certificates. The verifying party scans the QR code printed on the certificate and uses a password generated by the system to open the certificate being verified. This process also makes use of cryptographic algorithms to verify the hash. If the recomputed hash matches the initial hash, then the certificate is valid, otherwise it is invalid.

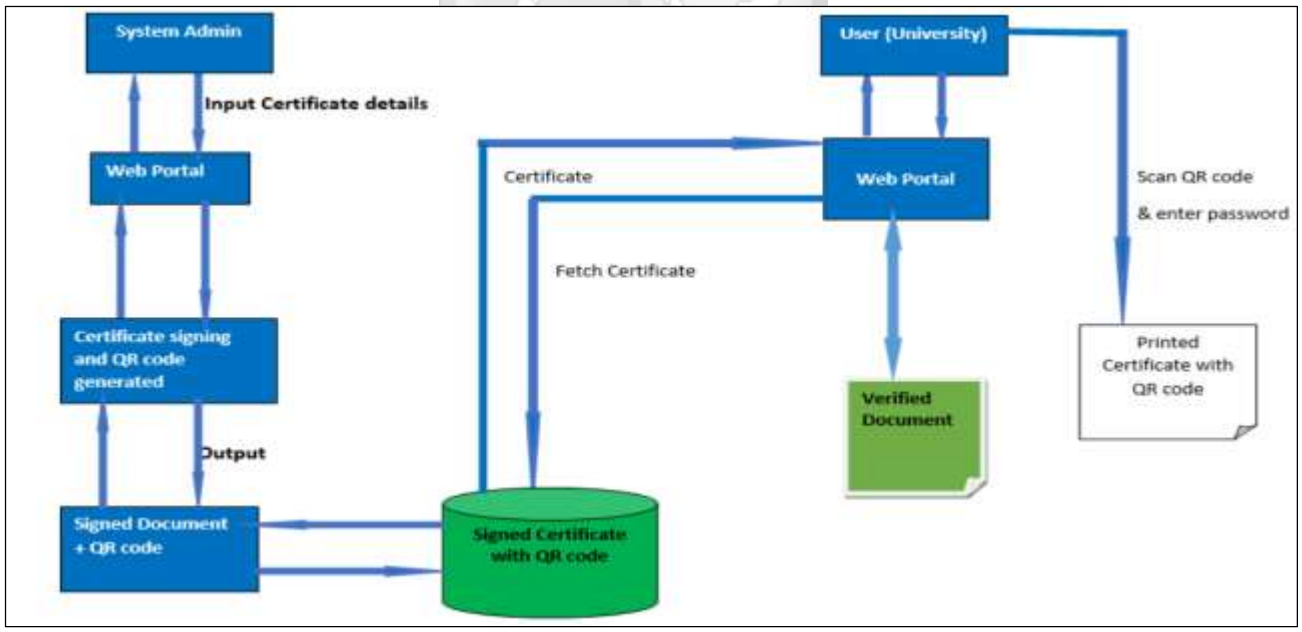
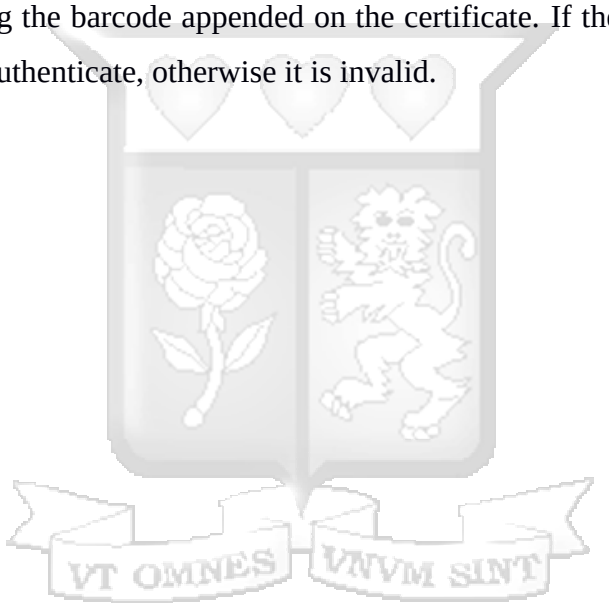


Figure 2.1: Certificate Verification Architecture Using Secure QR Codes (Kaibiru & Shibwabo, 2017)

Murthy et al. (2011) proposed an Elliptic curve digital signature algorithm (ECDSA) based method to control fake paper-based certificates. Their method involved combining cryptographic techniques alongside barcode technology to detect fake certificates. The paper assumed that the academic certificate issuing authority obtained digital certificates from a trusted Certificate Authority (CA) and that the public key of the academic certificate issuing authority was made publicly available. Hashes of certificate details were generated, digital signature generated and converted to barcodes and finally appending the barcode on the paper certificates. The verifier then takes the student's certificate details, generates a hash and compares it with the one generated from decrypting the barcode appended on the certificate. If the two hashes match, the certificate is considered authenticate, otherwise it is invalid.



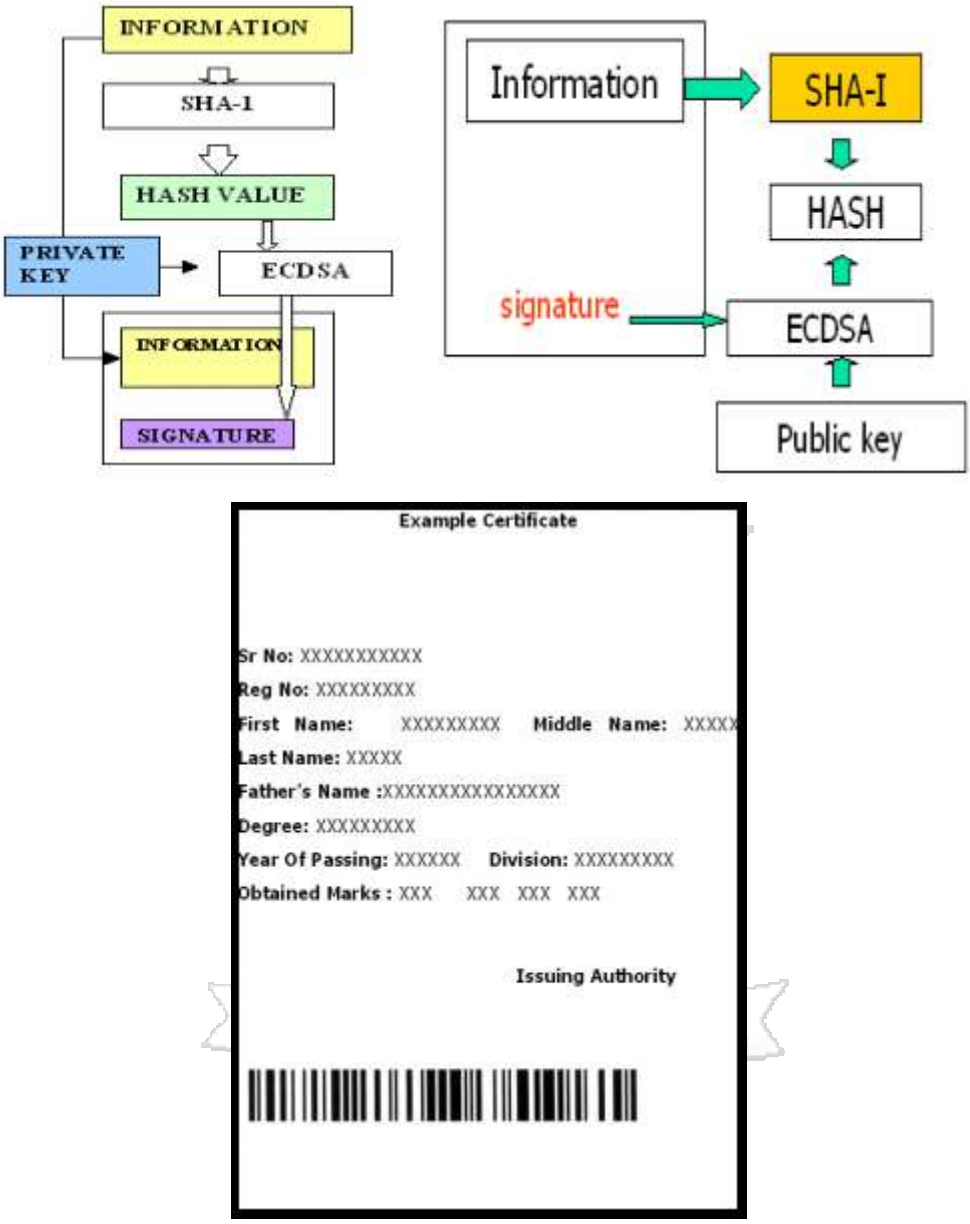


Figure 2.2: Use of ECDSA to Verify Academic Certificates (Murthy et al., 2011)

The Nigerian Universities Certificate Verification System was planned and executed to recover information from individual colleges through a web service, colleges are given format of the structure of the web service, in the proposed Optimization Universities are not restricted to simply the well-defined template, institutions can affix other applicable fields accommodating the assorted variety of the institutions. After expending the web services for information recovery from the Universities, the information is retrieved in a JSON format, in the first plan the

system should process the information into a format that is compatible to a relational database system, and then copy it before it is analysed. The proposed adjustments utilize a NoSQL based database at the central database side and this helps in reducing the time for processing into the relational databases compatible format then parsed into the database (Yusuf, Boukar, & Muslu, 2017).

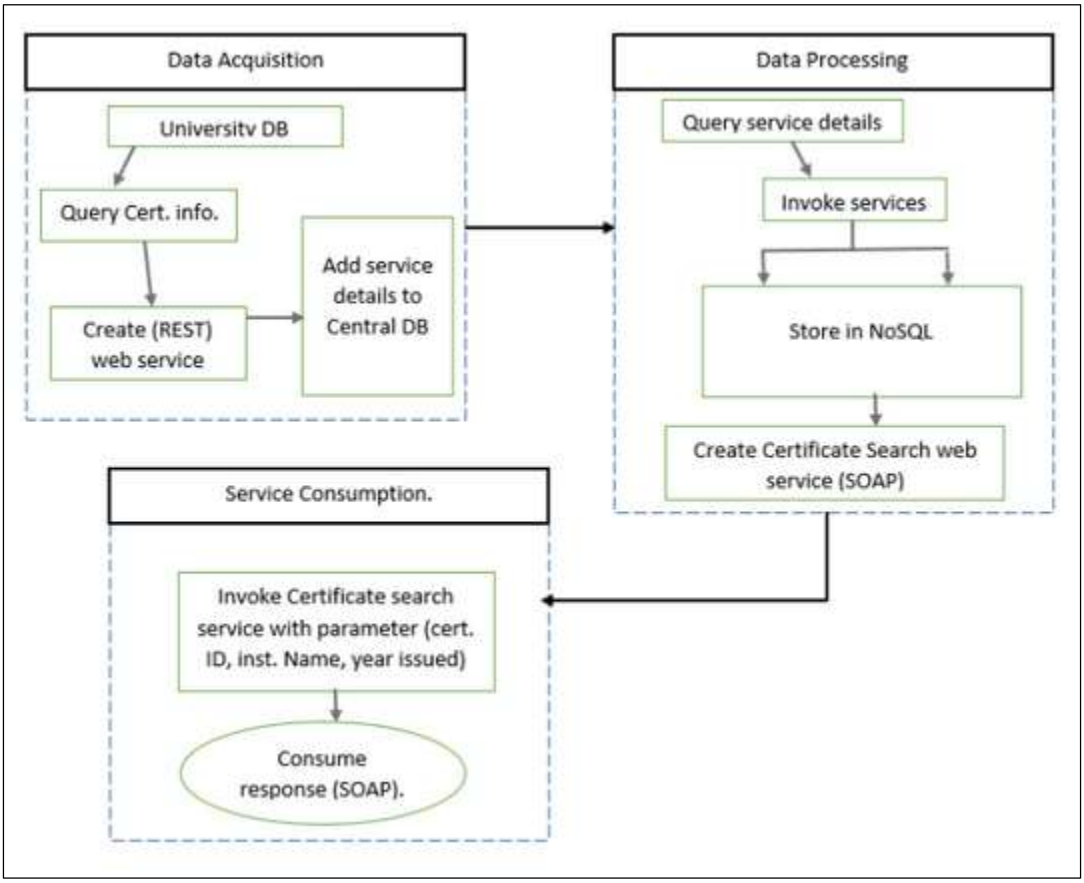


Figure 2.3: Nigerian Universities Certificate Verification System Architecture (Yusuf et al. 2017)

Ochieng (2016) proposed a client server architecture for verifying University certificates using QR codes. The client side consisted of the mobile application that contained the certificate verification application and a QR scanner while the server side hosted the centralized database of all certificates. The whole idea was for Universities to maintain a central database of certificates. A user of the system is required to select the University with whom they wish to verify a certificate from a list provided. The unique identifier of the certificate is then taken by scanning

the QR code on the certificate and the details submitted to the centralised database over an Internet connection. Details of an existing certificate will automatically pope on the client application and the certificate will be viewed as a genuine copy.

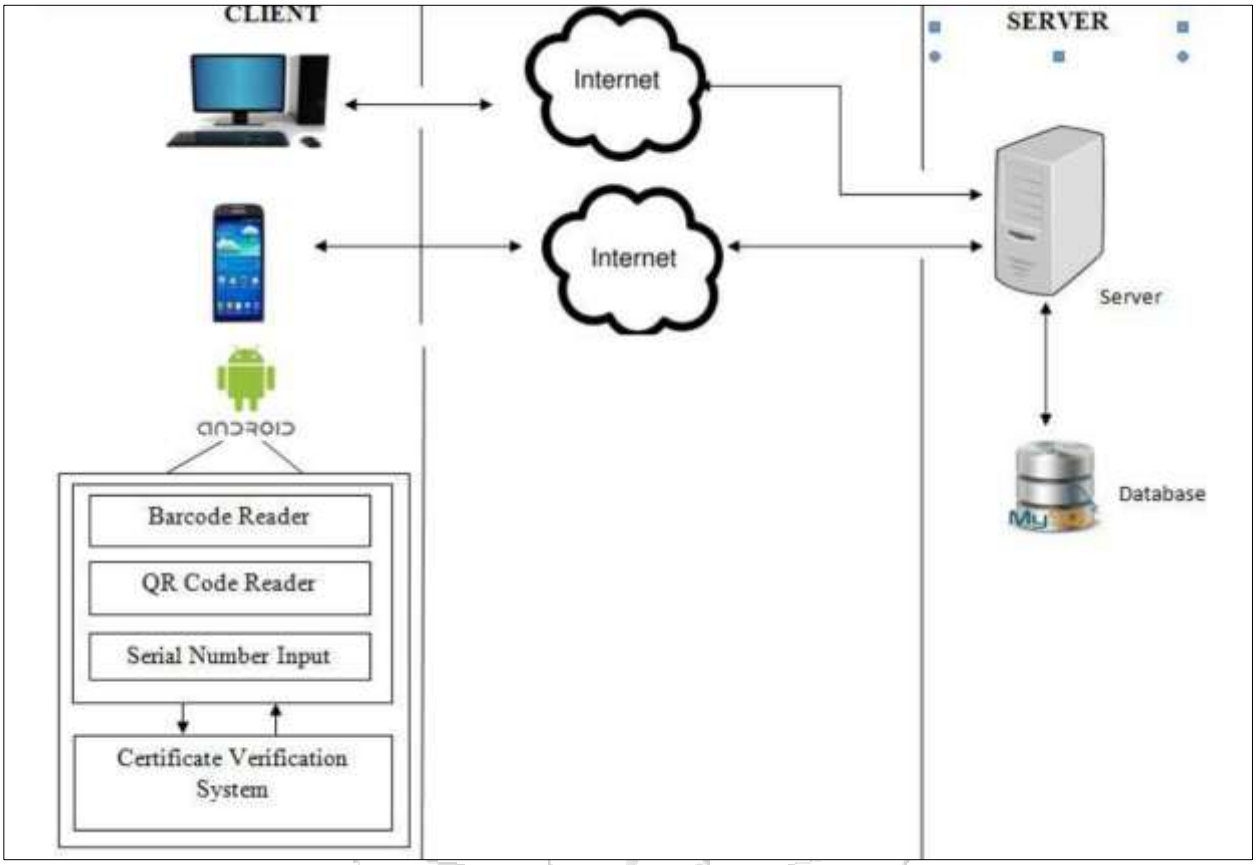


Figure 2.4: Client-Server System Architecture for the Certificate Verification Application (Ochieng, 2016)

2.3.3 Advantages and Disadvantages of Barcode for Verification of Academic Certificates

Some of the advantages as discussed by (Ochieng, 2016) include:

i. Barcodes can be embedded on almost everything making the easily adopted.
ii. In comparison to other tags such as Radio Frequency Identifier (RFID), barcodes have a higher degree of accuracy.

The disadvantages of barcodes include:

i. Each barcode must be scanned individually making the process labour intensive.
ii. If a barcode embedded on a product is damaged or lost, then it will not be possible to scan the product.

iii. The security of barcodes is questionable as they can be easily forged.

The reviewed techniques for verifying academic certificates include use of barcodes, QR codes, manual processes and online verification. It is very evident that the manual verification process is time consuming and, in most cases, prone to errors. The use of barcodes and web-based technologies have their advantages but the blockchain technology surpasses them due to its immutable and provenance nature.

2.4 Blockchain Technology

A blockchain can be viewed as an incorruptible distributed ledger consisting of all transactions made or simply digital events executed and shared among a group of people. Once information is entered into the database it can never be changed. Blockchain allows people to trust each other and transact within large peer-to-peer networks without centralized management. Each digital transaction in the thread is referred to as a block and the blocks are linked together to form a chain of blocks typically known as the blockchain. The contents of the blocks can either be agreed upon earlier or randomly generated by users of the blockchain. Once a block has been added by consensus among participants, it cannot be removed or altered, even by the original authors making the blockchain immutable. Other advantages of the blockchain include provenance; the origin of an asset is known and can be traced, consensus; all participants confirm that a transaction is valid coming to agreement on the updated state of the ledger and finality; assurance of having an up to date copy of the ledger (Pilkington, 2016; Sharples & Domingue, 2016).

To ensure confidentiality, integrity and non-repudiation, blockchain makes use of public cryptographic mechanisms. This involves the use of public and private keys, where the private key is used to encrypt data while the public key is used for decryption. The two cryptographic keys also make it possible for digital signatures to be generated allowing the receiver to verify whether data was changed on transit. Cryptography and distributed computing are the two major building blocks of the blockchain technology (Kosba, Miller, Shi, Wen, & Papamanthou, 2016; Zhou & Chow, 2009).

There exist two categories of blockchains. These are public and private blockchains. Public blockchain networks are basically permission-less networks where any node willing to participate can join at their own convenience. Permission less blockchain require use of specialized algorithms to ensure consensus of the network (Underwood, 2016). Private networks on the other hand are permission oriented and participants can only join through an invitation or upon request. These networks don't require any algorithm to come to a state of consensus. This study will seek to develop a blockchain application that is based on private blockchains.

2.4.1 Blockchain Transaction Process

Every participant in a blockchain network has a pair cryptographic keys that are used for encryption, decryption and signature verification purposes. Every transaction is sent as broadcast to all nodes in the network and only recorded in the public ledger after verification. A set of nodes referred to as validators are used for verification purposes to ensure that all nodes on the blockchain network come to a consensus on the state of the shared ledger. This is achieved by use of various distributed consensus algorithms namely, proof-of-work (PoW), proof-of-state (PoS), delegated-proof-of-state (DpoS) among others. The algorithms are built on a simple mechanism that for a node to be trusted, it must contribute something valuable to the network.

In the PoW algorithm, there exist a predefined value used by all nodes to verify validity of a transaction. Each miner first creates a block consisting of the unconfirmed transaction, a nonce and tries to generate a hash of the block. The hash depends not only on the transaction but the previous transaction hash. This is usually a compute-power-intensive competitive process where the node with more compute power stands a high chance of hashing the unconfirmed transactions. If the output of the hash matches the predefined value, the block is broadcasted to the network and validated by other peers after which it is added to the distributed shared ledger. At this point, the transaction in the ledger can never be removed or altered. The bitcoin uses this distributed consensus algorithm (Turkanović, Hölbl, Košič, Heričko, & Kamišalić, 2018; Zignuts Technolab, 2018).

Taking A as a sender, if entity A wants to add a document to the digital ledger, A will have to possess a pair of related keys namely private and public key. The private key is used for

encryption and digital signature generation whereas the public key is used for decryption. For the document to be sent, a hash of the document will have to be generated using a hashing algorithm

Secure Hash Algorithm (SHA) -256. The hashed value and private key are then fed into a signing algorithm to produce the digital signature which is sent along with the encrypted document and public key. The receiver will then use the same hash algorithm as the sender together with the public key supplied to generate a hash from the received document and digital signature. The generated hash is compared with the one received from A and if they match the validity of the document sent by A is approved and added to the public ledger. If there were to be any slight change on the document sent by A, the hash generated by the receiver will be different and thus the document will be rejected (Jeppsson & Olsson, 2017).

2.4.2 Advantages and Disadvantages of Blockchain Technology

Golosova and Romanovs (2018) outline the advantages and disadvantages of blockchain technology.

i. **High levels of security**. A consensus among participating blockchain nodes must be agreed upon before any transaction is committed to the ledger and stored inform of a hash of the previous block. This makes it impossible for information stored on the ledger to be tampered with.
ii. **Provenance**. Blockchain provides an audit trail of assets allowing people to see where the different parties that were involved in exchanging the asset. This enables the authenticity of an asset to be verified eliminating fraud case.
iii. There are also **reduced business cost** as organization no longer have to engage with trusted third parties for verification of documents and processes. The blockchain is trustworthy by itself.
iv. There is **increased transparency** as all participating entities on the blockchain network share a common ledger which is only updated after a consensus is agreed upon.

Golosova and Romanovs (2018) discuss the major disadvantages of blockchain technology as:

i. Extremely High Compute Power is required for processing blockchain transactions due to complex algorithms that need to be run during these processes.

ii. Not everything can be solve using blockchain technology. It is very important to evaluate the existing problem at hand before jumping into adopting the blockchain as a solution.

iii. The PoW and PoS algorithms are theoretically susceptible to some attacks though it will require huge compute power for these to be achieved. The attacks are: breaking of the cryptographic encryption algorithms and Distributed Denial of Service Attacks.

There exist weaknesses with the blockchain technology, but the strengths outweigh the weaknesses.

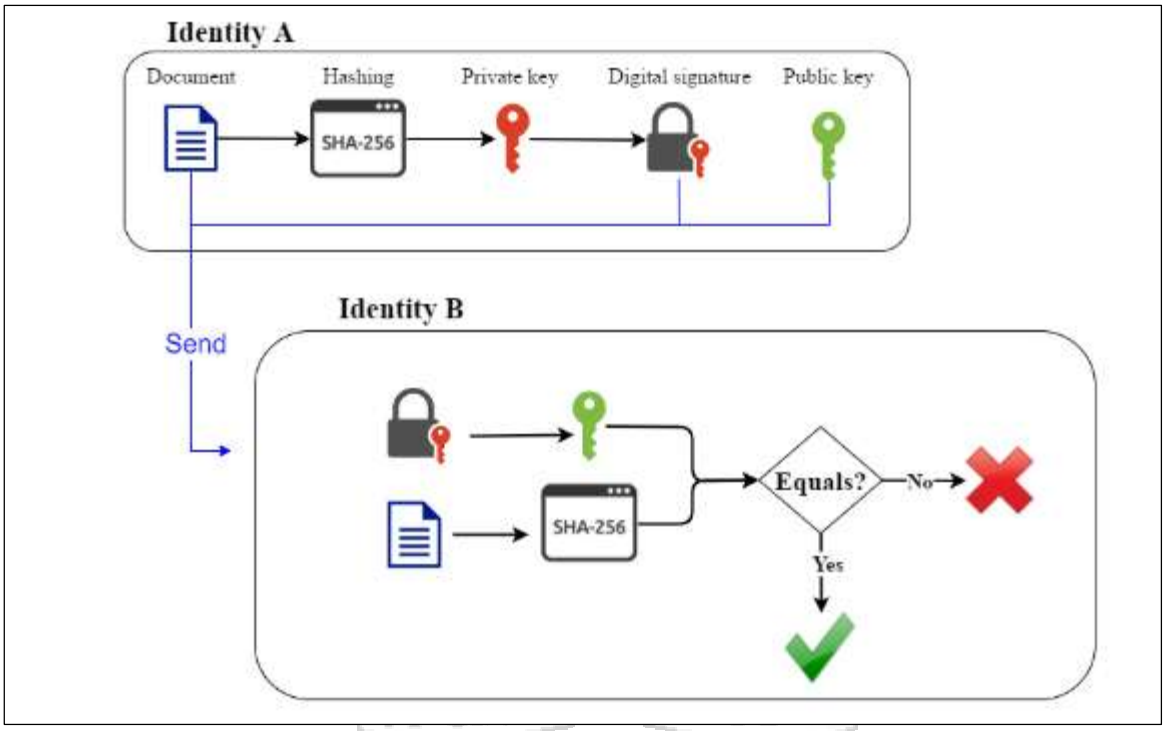


Figure 2.5: General Blockchain Transaction Process (Jeppsson & Olsson, 2017)

2.4.3 Applications of Blockchain Technology

The decentralized implementation of blockchain provides the required platform for development of alternate anti-counterfeiting mechanisms instead of relying on trusted third parties for authenticity of products. Brands, merchants and marketplaces can form a blockchain network with nodes storing information to validate authenticity of their products (Apte & Petrovsky, 2016).

The blockchain technology provides an alternative model to proof-of-existence and possession of legal documents. By leveraging the blockchain, a user can simply store the signature and timestamp associated with a legal document in the blockchain and validate it anytime using native blockchain mechanisms. Users can give proof of existence of their documents that can't be modified by a third party. This ensures privacy and security of user details is upheld (Lemieux, 2016).

Blockchain also facilitates the deployment of decentralized Internet of Things (IoT) platforms such as secured and trusted exchange of data within IoT domain nodes as well as record keeping. The blockchain acts as the ledger recording trusted records of all message exchanged (Zahid, Hussain, and Ferworn, 2016).

2.5 Ethereum Framework

Ethereum is an opensource, public blockchain development platform that was proposed by Vitalik Buterin in 2013 through a white paper. The development of the platform was founded through a crowd sale between July-August 2014 with participants buying the ether token value (Wood, 2014). The nature of this platform has it as being permissionless allowing anyone to join the network as no access control rules are defined.

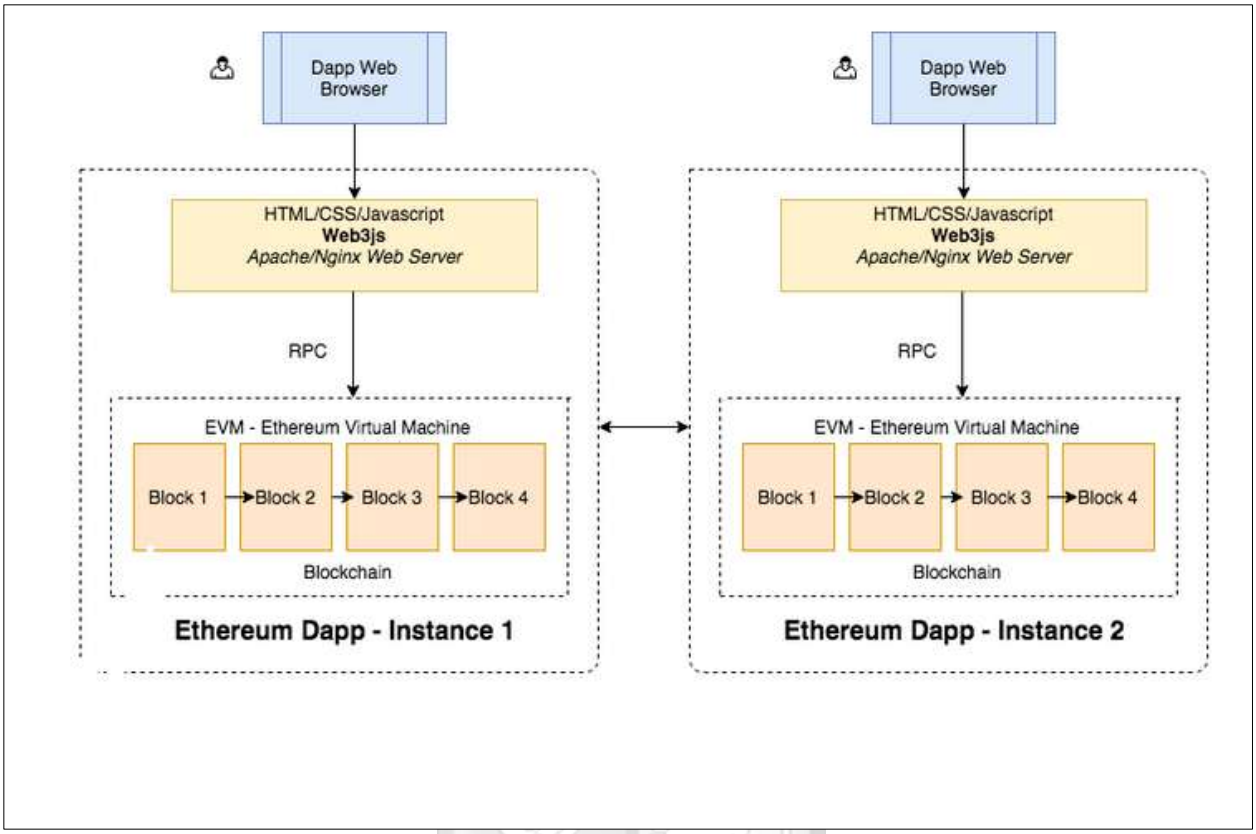


Figure 2.6: Ethereum Architecture (Zastrin, 2019)

An Ethereum blockchain has two major components namely the database and code. The database is responsible for storing transactions while the code is the logic part of the application. The applications deployed are build using inbuilt java script libraries that connect to blockchain nodes. To synchronize database across the entire blockchain network, Ethereum uses an algorithm referred to as Proof of Work (PoW) (Zastrin, 2019).

2.5.1 Ethereum Transaction Process.

Any function call is first converted into a raw transaction. To ensure that a valid user is the one executing a transaction, the transaction is signed using a valid private key corresponding to the account that initiated the transaction. The transaction is then validated locally on a user's local Ethereum node before being broadcasted to the network. Miner nodes then select transactions to be included in a block, validate the transactions and begin PoW. The miner node that first finds a valid block will add the block to the blockchain network and broadcast this new block to the entire blockchain network (Zastrin, 2019).

2.6 Hyperledger Framework

This is a Linux foundation project aimed at developing frameworks that make it easy to develop and deploy private business blockchain applications. The framework provides implementation of smart contracts, shared ledgers, membership services and a consensus mechanism (Hyperledger, 2018).

The Hyperledger Composer and Fabric frameworks were the chosen frameworks for the development of the blockchain solution. Hyperledger Fabric hosts the blockchain network which consists of smart contracts, shared ledgers, membership services and a consensus mechanism. The composer consists of three major components namely model file - defining class of assets, transactions, participants and events, script file: containing scripts that define transaction execution logic and the access file that defines permitted operations and access levels for users on the blockchain network. This assists in modelling the blockchain network and exposing it through Application Interface Programming (API) to the user application, invoking transactions to interact with the blockchain network and provide integration with existing systems (Hyperledger, 2018).

2.6.1 Hyperledger Fabric Transaction Process

A typical Hyperledger fabric network consists of three types of nodes. These include client, peer and ordering-service nodes. A client is an entity that acts on behalf of an end user and is responsible for invoking transactions by connecting to any peer of its choice. Peers on the other hand are responsible for holding the ledger and the world state. The ordering service is a communication fabric that ensures transactions are arranged in a timely order i.e. first come first serve basis. In blockchain technology, a transaction is an event that describes the exchange of assets between participants while assets are anything that can be owned or controlled to produce value.

A client creates a transaction and forwards it to any endorsing peer of its choice on the blockchain network in a message format referred to as a proposal. The endorsing peer then confirms the client's signature and if valid simulate a transaction by executing a chain code referred to by the transaction. The peer forwards the transaction internally to its endorsing logic

which signs the proposal and send it's back to the client. If the endorsing logic refuses to endorse the transaction, a reject message will be sent back to the client. The submitting client waits to receive a good number of endorsement signature from the peers and finally broadcast the transaction to committing peers through the ordering service. Committing peers are used to validate each transaction in the block and if valid, they commit the block to the ledger (Morris et al., 2018).

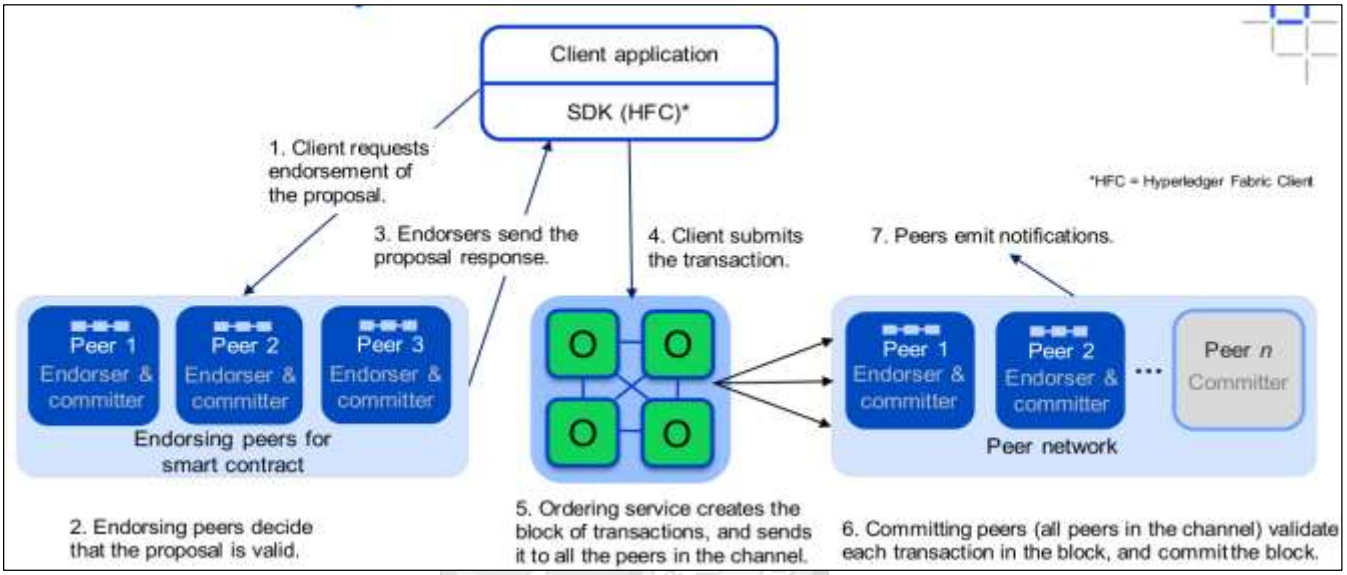


Figure 2.7: Hyperledger Fabric Transaction Endorsement Process (Hyperledger, 2018)

2.7 Test Driven Development

A test-driven development approach was be adopted in this research. This involved outlining the business logic and defining test cases relating to the following objectives:

1. Testing logic files that define system transactions and provide concrete implementations.
2. Testing query files.

To achieve the mentioned objectives, a combination of functional and scenario-based tests were carried out (Morris et al., 2018).

2.8 IBM Blockchain Platform

The IBM blockchain platform is based on the Hyperledger framework and is offered as a service in two categories. These are the starter plan which provides a test environment for developers and the enterprise plan that is used in production environment. is available at a cost that offers a highly secure environment through which authenticated members can quickly define assets and create solutions for modifying and exchanging them. The cloud platform is organized into three major layers namely; data layer, business application layer and the presentation layer. The starter plan offers high availability, scalability and resilience while incorporating tools that simplify administrative tasks. However, one is not able to access the backend of the platform and this possess a challenge for a researcher who is more comfortable working from the command line interface. This package is available free of charge to students whose universities partner with IBM.

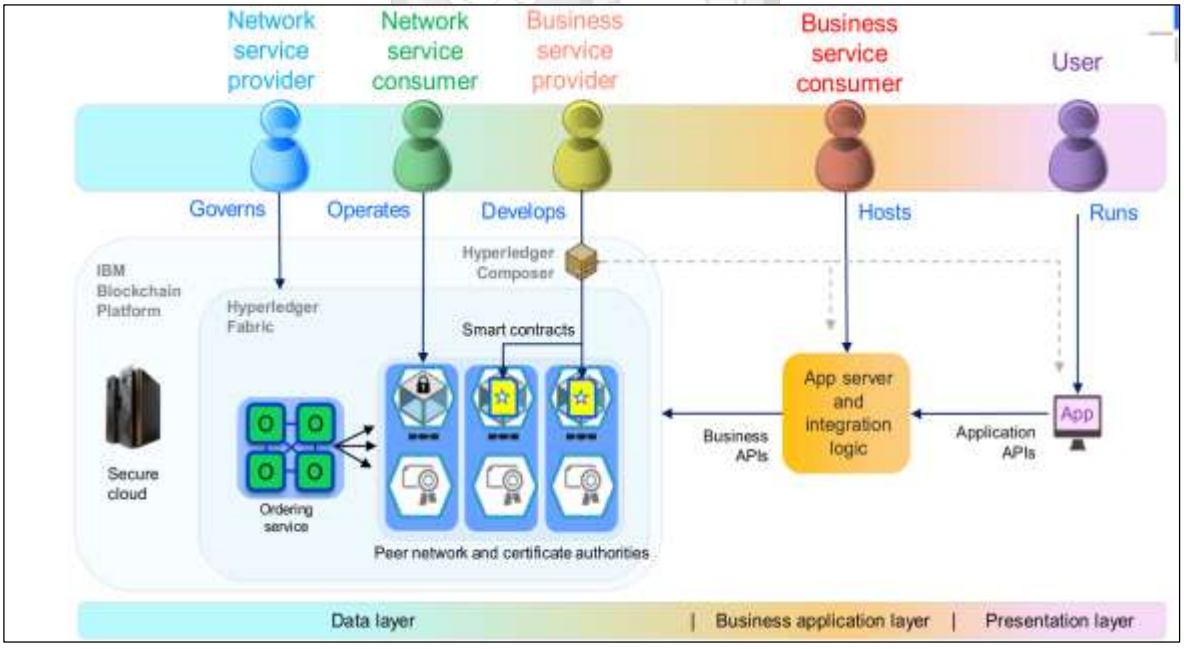


Figure 2.8: IBM Blockchain Platform Network Architecture (Hyperledger, 2018)

2.9 Local Development Environment

The Hyperledger composer and fabric framework can be installed on a local host machine and offer similar results to the IBM blockchain platform. It however gives more control to a developer since one will have access to both the Graphical and Command Line Interfaces of the platform. This was the preferred development platform as it offered more flexibility to the researcher. A server was provisioned on a cloud platform and installed with the necessary components. The prerequisite for installing the framework on a local host machine are:

1. Operating Systems: Ubuntu Linux 14.04 / 16.04 LTS (both 64-bit), or Mac OS 10.12
2. Docker Engine: Version 17.03 or higher
3. Docker-Compose: Version 1.8 or higher
4. Node: 8.9 or higher (note version 9 and higher is not supported)
5. npm: v5.x
6. git: 2.9.x or higher
7. Python: 2.7.x

CHAPTER THREE

RESEARCH METHODOLOGY

3.1 Introduction

This section examines the research approach used during the study. This includes steps, procedures and methodologies used to carry out the study and developing the solution.

3.2 Research Design

The research utilised mixed methods comprising of descriptive research and prototyping in order to achieve the research objectives.

A descriptive research approach was used to address objective one, two and three while making use of qualitative data to analyse the need for a blockchain capable system for purposes of verifying academic certificates. The prototyping approach was used to address objective four of this research.

3.3 Target Population

Asiamah, Mensah, and Oteng-Abayie (2017) define target population as a group of individuals or participants with specific attributes of interest and relevance to a research study. This research targeted a study population constituting of university staff and employer companies, comprising of individuals across various departments such as academia, administration, human resource and Information Technology fields. The universities were targeted as they issue and verify academic certificates while employer companies verify potential employee certificates.

3.4 Sample Size

A total of 10 respondents were involved in this study who were in one way or another involved in certificate verification. Respondents were selected using stratified random sampling techniques since the study targeted a specific group of respondents. These comprised of university employees and recruitment agencies within Nairobi and its environs.

3.5 Data collection

This research used both primary and secondary data sources. Primary data was collected using informal interviews to the selected participants together with observation. This were conducted with the sole purpose of identifying the current methods of certificate verification in different places i.e whether manual or automated and to better understand the existing challenges in academic certificate verification process. Secondary data was collected through reviewing literature from various authors and non-commercial databases. The data collected informed the basis for developing the prototype solution for the research.

3.6 Data Analysis

This involved organizing the data collected in manner which was clearly understood. Due to the nature of research questions and specific objectives, this research used descriptive techniques to analyse data. Data collected was qualitative data and was therefore analysed and presented using pie charts and bar graphs to provide a clear interpretation of the findings.

3.7 Systems Development Methodology

The final product of this study was a software application prototype that demonstrated the key system functionality and proof of concept. Rapid Application Development (RAD) prototyping was the approach selected to develop the application. This was the most appropriate approach as it laid emphasis on iterative manner of developing software with the final product being a prototype. This simply meant that the researcher had to go through several iterative steps from requirements gathering, rapid system design, construction of the protype, review of the protype constructed and finally reworking the prototype through the iterative steps until the general research objectives was realized. Figure 3.1 shows the RAD model.

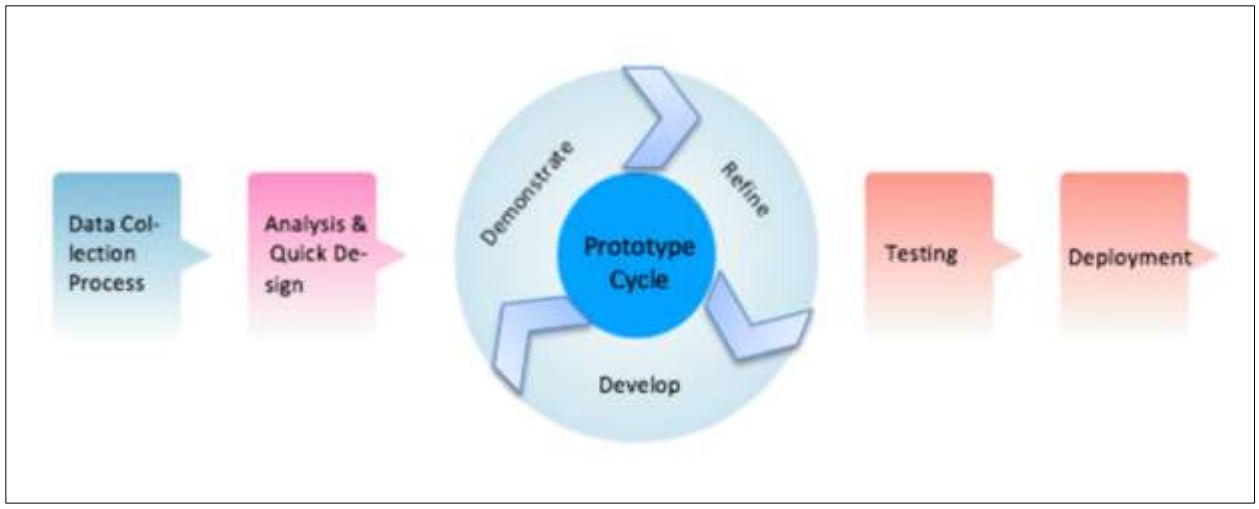


Figure 3.1: RAD Model (Tongkaw, Inkaew, & Tongkaw, 2019)

3.7.1 Requirements Gathering

In this phase, the scope of the project and the application's functional and non-functional requirements were identified by the researcher to facilitate the commencement of the next phases. The functional requirements are sometimes referred to as business processes.

3.7.2 Rapid System Design

The system design phase involved dividing the business process from the requirements gathering phase into two groups namely system inputs and system outputs. The system processes were modelled using appropriate software modelling techniques. These involved using data flow diagrams, use case diagrams, and sequence diagrams showing how actors interacted with the system.

3.7.3 Rapid Prototype Construction

The logical and physical designs identified enabled the researcher to embark on actual system implementation. The initial prototype was developed by making use open source code. Various challenges were encountered during the process especially on the implementation environment. Source code failed to compile at some point forcing the researcher to shift to a different hosting platform. A few changes were made, and everything went on well as expected. The application

was later developed on a cloud based virtual server and Hyperledger Composer and Fabric were used to develop the blockchain application.

3.7.4 Prototype Review

In the prototype review stage, feedback of what is good and what is working was received. This was made possible by subjecting the system to functional, compatibility and integration tests to determine whether everything was in line with the identified requirements. Based on the feedback received, the prototype was further modified to until a product that realised the general objective of the study was developed.

3.8 Ethical Issues

The data collected from respondents was treated with high degree of confidentiality and was solely used for the intended purposes. The researcher did not force participant to participate in the survey. Literature obtained from sources such as journals was referenced and cited appropriately by the researcher.

CHAPTER FOUR

DATA ANALYSIS AND SYSTEM DESIGN

4.1 Introduction

This section presents the findings from data collection processes as well as describing the various functional and non-functional system requirements. High-level system architecture and system design are presented using use case, sequence and context diagrams.

4.2 Data Analysis

Establishing the methods used and challenges encountered in certificate verification was of interest to the researcher with the sole purpose of determining the existence of the research gap. The researcher reviewed literature from various scholars and conducted informal interviews to collect the data. Ten respondents who were willing to participate in the study were identified and informal interviews conducted. Findings presented in this section are therefore based on the information gathered during the interview sessions and reviewed literature.

4.2.1 Interview Questions

Figure 4.1 shows the questions asked during the interview sessions together with the raw feedback from the respondents.

Question	Category	No. of Responses
Which department are you affiliated with?	Administration office	2
	Human Resource	2
	Information technology	5
What method is used by your organization to verify academic certificates?	Manual	8
	Automated	1
Do you have any particular parameters that uniquely identify certificates issued by your organization?(e.g University Name, Logo, Watermark etc). Please list the features.		**Most made use of the mentioned features**
Have you ever had incidences of falsified academic certificates presented to your organization by candidates?	Yes	3
	No	6
Do you think a technology that holds immutability, provenance and privacy would strengthen academic certificate verification process at your organization?	Yes	9
	No	0

Figure 4.1:Interview Questions

4.2.2 Response Rate

The interviews were conducted through phone conversations, one on one sessions and by administering short questions through email for the respondents to give their input. Nine (9) Out of the ten (10) identified respondents gave their feedback. This represents a 90% response rate as shown in Figure 4.2

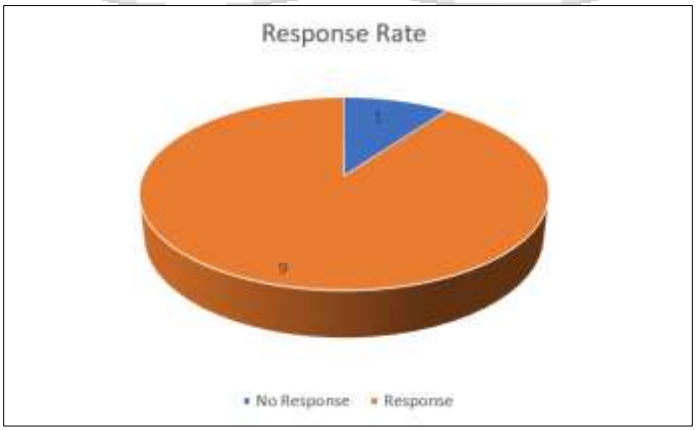


Figure 4.2: Response Rate

4.2.3 Certificate Verification Methods

Figure 4.3 shows that Eight (8) respondents use manual certificate verification methods at their organizations while only one (1) had an automated system in place. From the reviewed literature government agencies in Kenya and other organizations also still relied on manual processes of academic certificate verification.

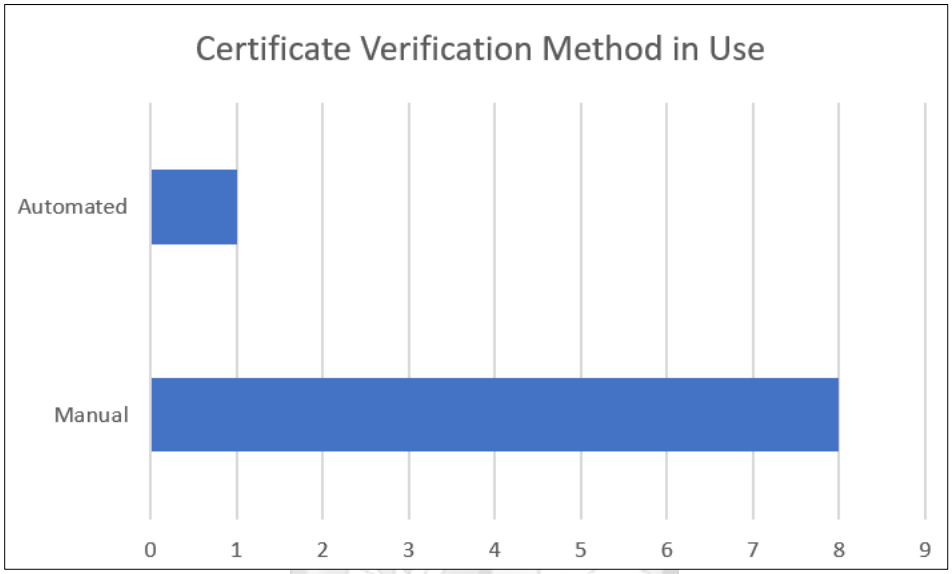


Figure 4.3: Responses on Certificate Verification Methods Used

4.2.4 Challenges Encountered in Certificate Verification Process

All the respondents indicated that they experienced challenges in the verification processes and more so those who used manual processes which they pointed out as being tedious, time consuming and prone to errors.

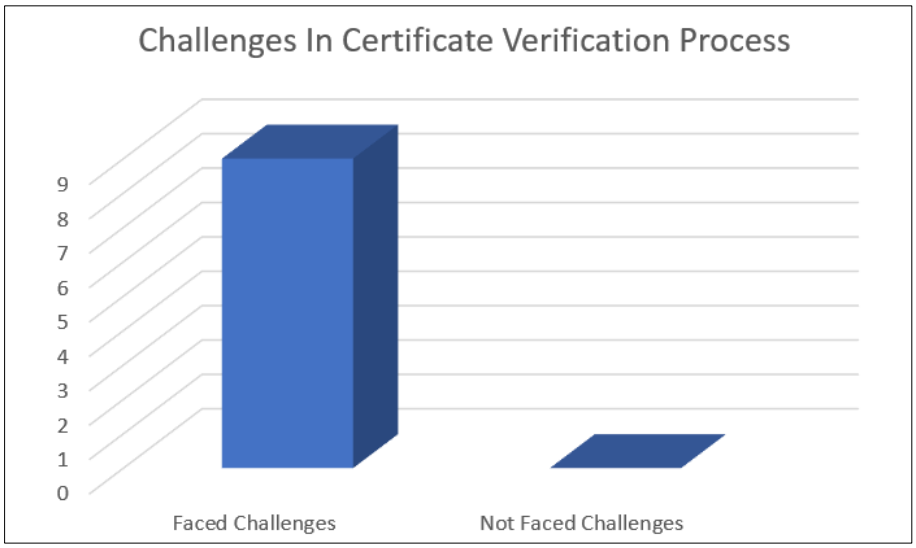


Figure 4.3: Respondents who Experienced Challenges in Certificate Verification

4.2.5 Views on the Proposed Technology for Verification of Academic Certificates

The nine respondents were all in agreement that blockchain technology would come in handy in solving the current problems faced during the certificate verification process as shown in Figure 4.5.

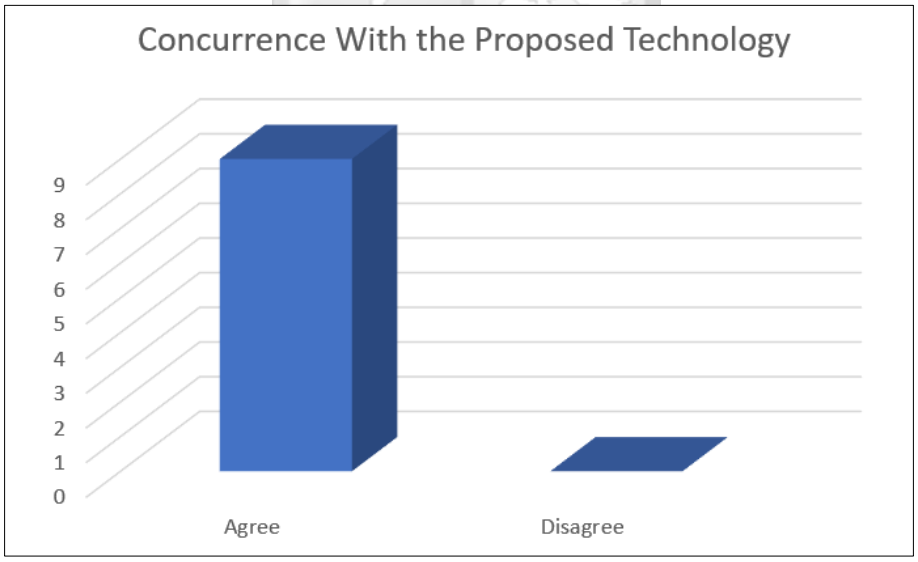


Figure 4.4: Respondents Concurrence on the Use of the Proposed Technology

4.2.6 Data Analysis Summary

From the findings presented, it is evident that certificate verification process remains a challenge to many organizations who still rely on manual verification methods. The respondents were also in agreement that the use of a technology that offers immutability, provenance and security would go a long way towards countering academic certificate falsification.

4.3 Requirements Analysis

The requirements analysis phase involves identifying the functional and non-functional system requirements and clearly stating them. This was to help during the testing phase to ensure that the system was functioning as expected.

4.3.1 Functional Requirements

Functional requirements define what a system must be able to successfully accomplish in terms of input fed to it. The functional requirements for blockchain application included:

i Create account

All users must first be created for accounts to be able to use the system.

ii. Login

All users with accounts created should be able to access the system by entering correct system credentials.

iii. Enrol users

A Hyperledger Fabric Administrator should be able to add users who can interact with the fabric at various levels.

iv. Delete users

A blockchain Network Administrator should be able to delete users from the system

v. Generate certificate

A blockchain Network Administrator should be able to generate user certificate to be stored on the ledger.

vi. Verify certificate

A registered user should be able to query the ledger to cross check a student's certificate.

4.3.2 Non-Functional Requirements

These are global constraints on various services offered by a system and don't affect how the system works. They include:

i. Reliability - The system should be available to users whenever they need. This implies zero or very minimal downtime.

ii. Usability - The system should be simple to use with no steep learning curve.

iii. Scalability - The system should be modelled such that more modules can be easily added

iv. Security - The system should incorporate all major aspects of security i.e confidentiality; by ensuring only authorised people have access to information and integrity to ensure that information stored on the system is not tampered with.

4.4 Process Modelling

Feiler and S. Humphrey (1993) describe process modelling as an abstract representation of architecture, design or definition of software processes. This section shows the different process models representing the logical modelling of the blockchain application. These include use case, sequence and context diagrams.

4.4.1 Use Case Diagram

Use case diagrams are behavioural in the sense that they describe a set of actions that a system performs in response to input from one or more external users/actors. Figure 4.6 shows the interaction between users and system processes. The main actors are employers/student, Blockchain fabric Administrator and Blockchain Network Administrator. The major use cases are also described in detailed. Table 4.1 and Table 4.2 show the use case descriptions of adding

an Asset/participant and querying the system to obtain certificate details respectively. An asset in this case is a certificate while a participant is an employer/administrator.

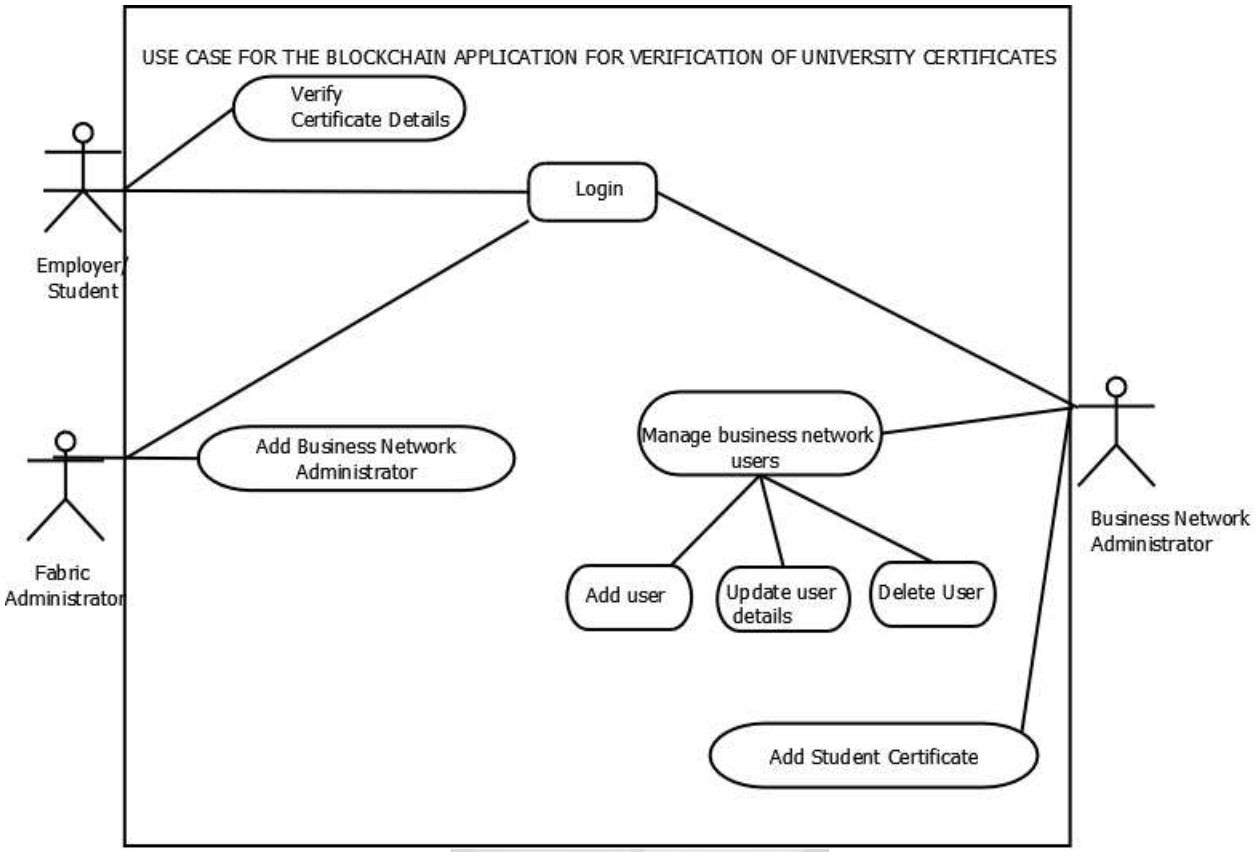


Figure 4.5: Use Case Diagram for the Blockchain Application

4.4.2 Use Case Description

Table 4.1: Use Case Description of Adding an Asset/Participant

USE CASE ID	**UC1**
Title	Adding an Asset/Participant
Description	User access the asset/participant menu on the graphical user interface of the application.
Actor(s)	Faculty Administrator.
Pre-conditions	User must have logged in to the system and launched the application's Graphical or Command line interfaces.
Main Success Scenario	1.User selects "Create Asset/participant" button. 2.User then enters all the data associated with an asset or participant. 3. User clicks on the confirm button. 4. The transaction details and asset/participant details are added to the blockchain and state ledgers respectively.
Alternative flow of events	System declines the data entered and returns an error message.

Table 4.2: Use Case Description of Querying the System to Obtain a User's Certificate Details

USE CASE ID	**UC1**
Title	Verify Certificate details
Description	User access the GET/Query menu on the graphical user interface.
Actor(s)	Employer.
Pre-conditions	User must have logged in to the system and launched the application's Graphical or Command line interfaces.
Post conditions	A user's certificate details must have been entered to the blockchain ledger.
Main Success Scenario	1.User selects "Get certificate/Post". 2.User then enters a student's certificate identification number. 3. User clicks on the submit button. 4. The certificate details are displayed on the graphical user interface or command line interface.
Alternative flow of events	System declines the data entered and returns an error message

4.4.3 Sequence Diagrams

Sequence diagrams are used to show the interactions among different system classes. They help to visualize and validate different runtime scenarios. The sequence diagrams described below show the interaction between actors and main system processes. Figure 4.7 and 4.8 show sequence diagram for adding an asset/participant and certificate verification respectively while Figure 4.9 shows the main sequence diagram for a transaction process on the blockchain network.

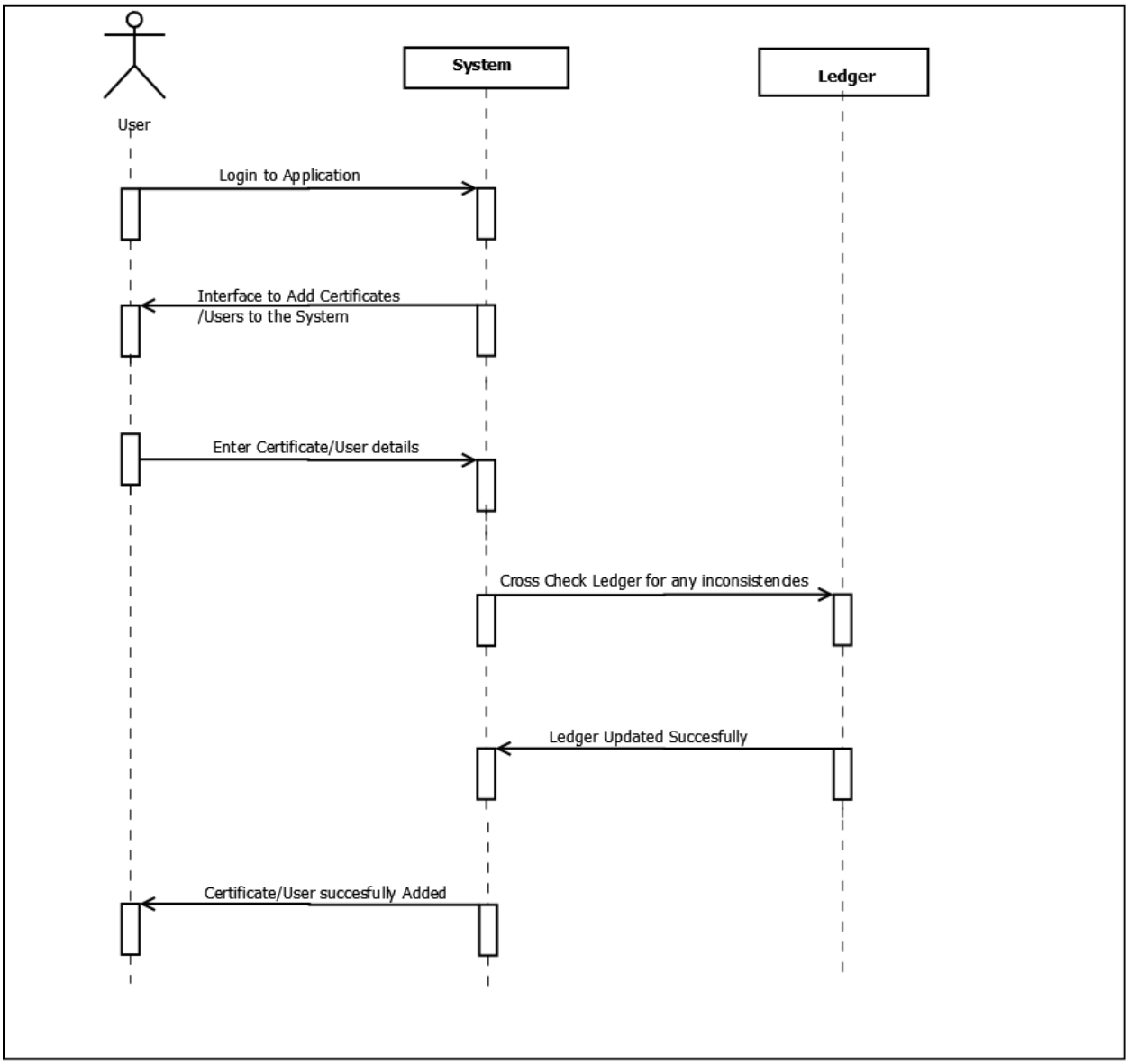


Figure 4.6: Sequence Diagram for Adding an Asset/Participant

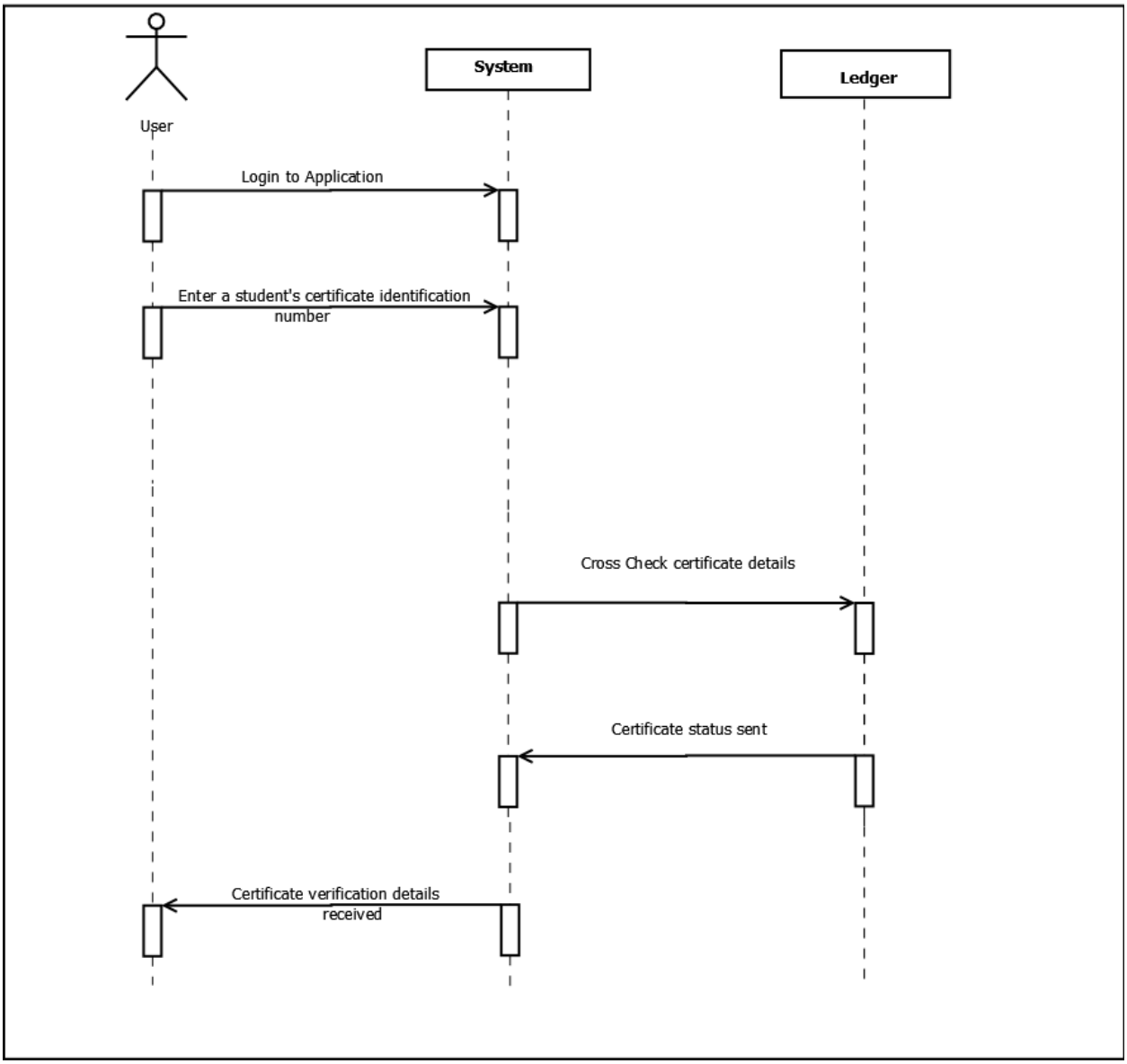


Figure 4.7: Sequence Diagram for Verifying a User's Certificate

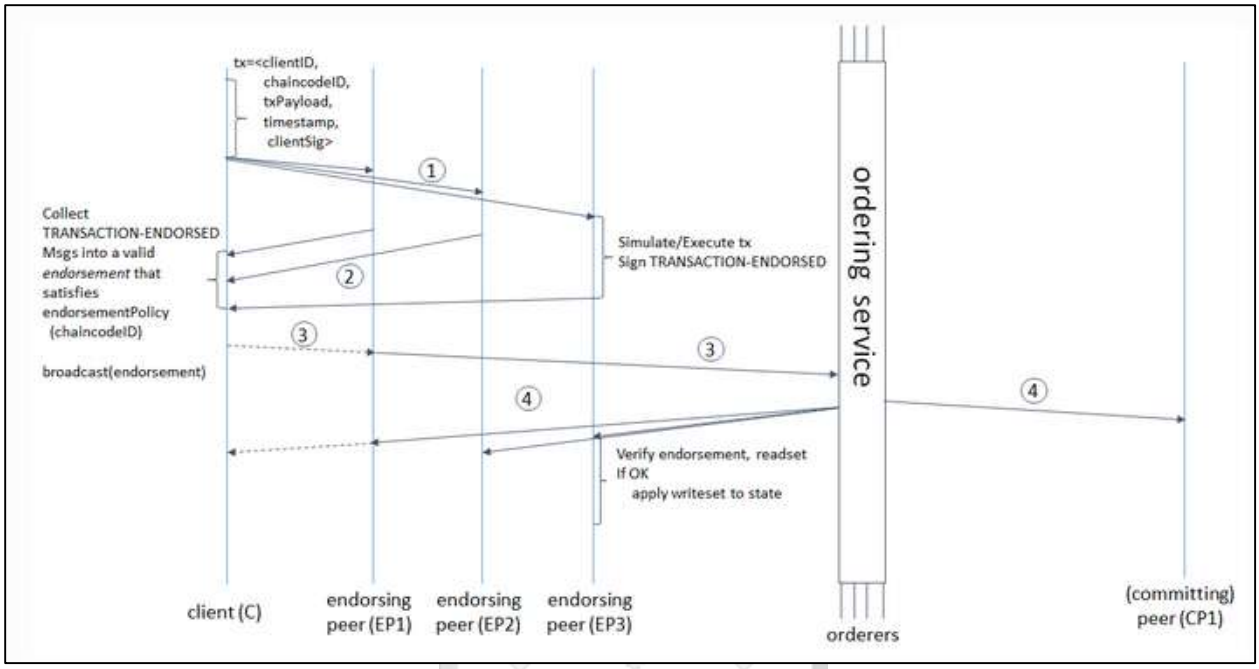


Figure 4.8: Sequence Diagram for a Transaction Process on the Blockchain Network

4.4.4 Data Flow Diagrams

Data flow diagrams (DFDs) are used to show the flow of data as it moves across a system in the form of inputs, outputs and data stores (Burge, 2011). This section shows the systems context diagram.

4.4.4.1 Context Diagram

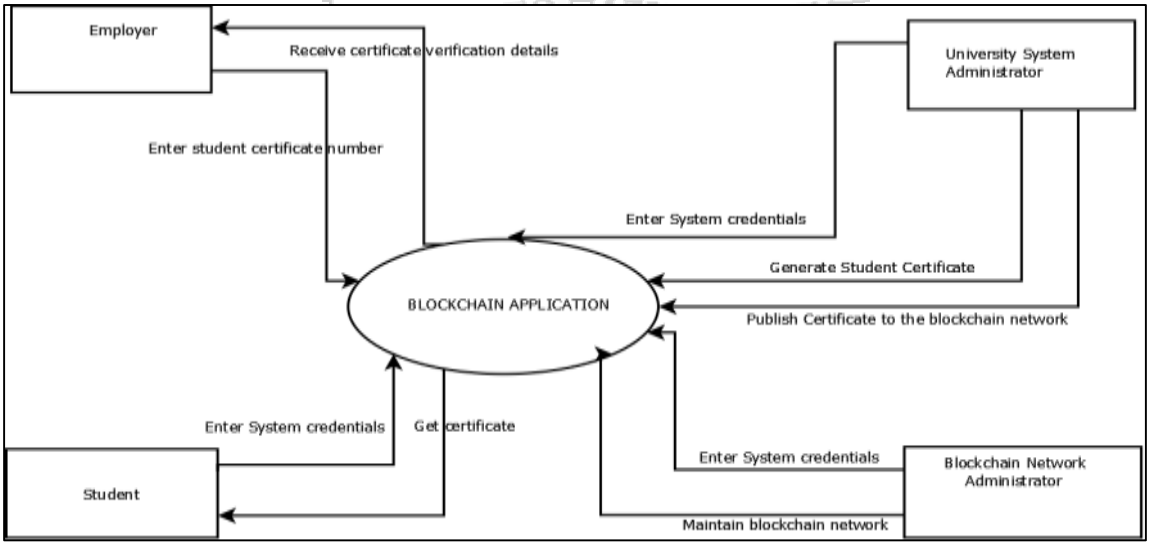


Figure 4.9: System Context Diagram

4.4.5 Data Stores

A peer node is responsible for holding the ledger that stores records of all transactions and current state of assets in the blockchain network. The ledger in this research is made up of two sets of data stores namely the blockchain and the world state. The blockchain chain is simply an immutable linked list of all transactions made on the blockchain network. The transactions include the action of generating a certificate, the process of adding system users just to mention a few. The transactions are stored as cryptographic blocks of hashes on the blockchain. The world state is modelled as a versioned key-value store (KVS), where keys are names and values are arbitrary blobs. Entries on the KVS are manipulated by the applications running on the blockchain by using put and get KVS-operations. The state is stored persistently and updates to the state are logged.

4.5 System Architecture

The system architecture shows the major system components while describing the services provided by each component.

4.5.1 High Level System Architecture

The whole idea of the research was for Universities to maintain a shared immutable ledger of academic certificates through replication of the shared ledger across the institutions. This was to be made possible through use of distributed ledger technologies and more specifically blockchain. The researcher proposed use of permissioned blockchain in order to have control of the network by ensuring that only legitimate institutions join the network.

Hyperledger Fabric was used to demonstrate the proposed application's functionality. The Fabric environment consists of several components namely:

i. Smart contracts responsible for stating the business logic as a piece of code and process transactions to determine if they comply with the existing business requirements.
ii. Ordering service which is responsible for the synchronization and ordering of transactions within the blockchain network.
iii. Nodes that hold, validated and endorsed transactions while maintaining a copy of the ledger referred to as peers. Smart contracts were executed here as well.

iv. Membership Service Provider (MSP) that maintain identities of users on the blockchain network.

A peer node is owned and maintained by an organization and in this research represents a university. A university can have more than one peer node to hold the ledger for redundancy purposes while maintaining its own independent environment. A group of universities could come together and setup peer nodes for hosting their institution academic certificates and share the information through channels setup on the fabric network. The universities then nominate individuals who will be responsible for managing the fabric network on matters such as creation of network channels and members. A university that wishes to participate in the setup network can then submit a request to join the network or receive an invite from an existing member to join the network. When all peers on the network communicate on the same channel, the ledger is replicated across all peers on the network causing them to have identical records.

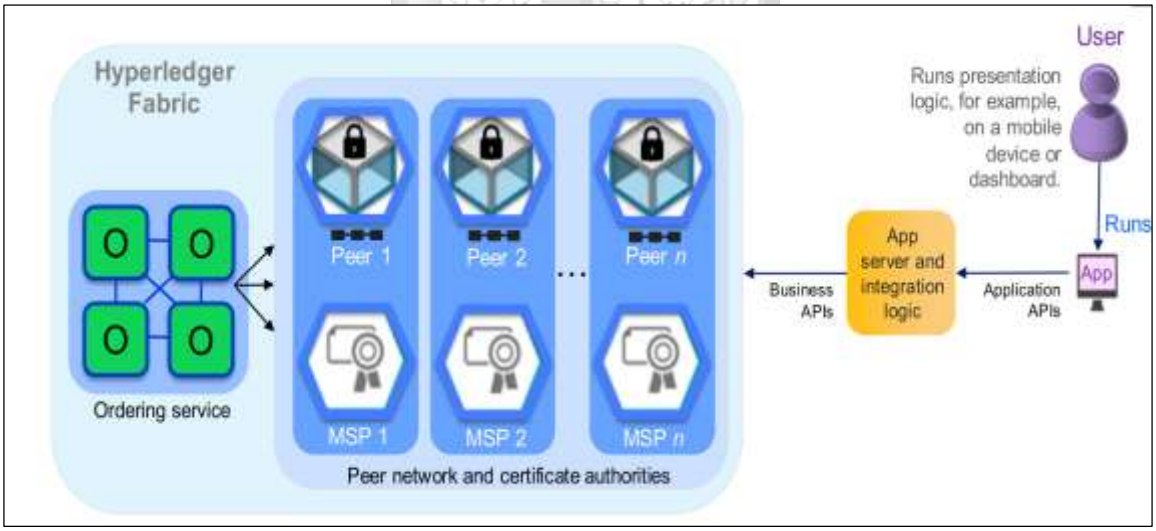


Figure 4.10: High level System Architecture (Hyperledger, 2018)

4.5.2 System Architecture of the Blockchain Certificate Verification Application

The system architecture for the developed application followed a client server model. In this architecture, tasks are partitioned between the providers of the resource known as servers and the consumers of the services known as clients (Ochieng, 2016).

In order to demonstrate the primary functionality of the proposed solution, a Hyperledger Fabric network with a single peer was setup. This represented a single organization. The deployed Hyperledger Fabric network was made up of a single organization called Org1. The organization used the domain name org1.example.com. All other components were also associated with the example.com subdomain. The system consisted of the following components:

i. Blockchain application user.

ii. Blockchain user application.

iii. Client node - This is an entity that will be acting on behalf of and end use by communicating with the blockchain through a peer.

iv. Hyperledger composer playground – A web tool for defining and testing Hyperledger composer models and scripts.

v. Hyperledger composer rest server – provides a graphical user interface to allow querying the fabric network and generation of Rest API.

vi. Hyperledger fabric network consisting of membership service provider, smart contract, single peer and orderer and a Certificate Authority (CA). Certificate Authority is responsible for issuing cryptographic certificates to all system components.

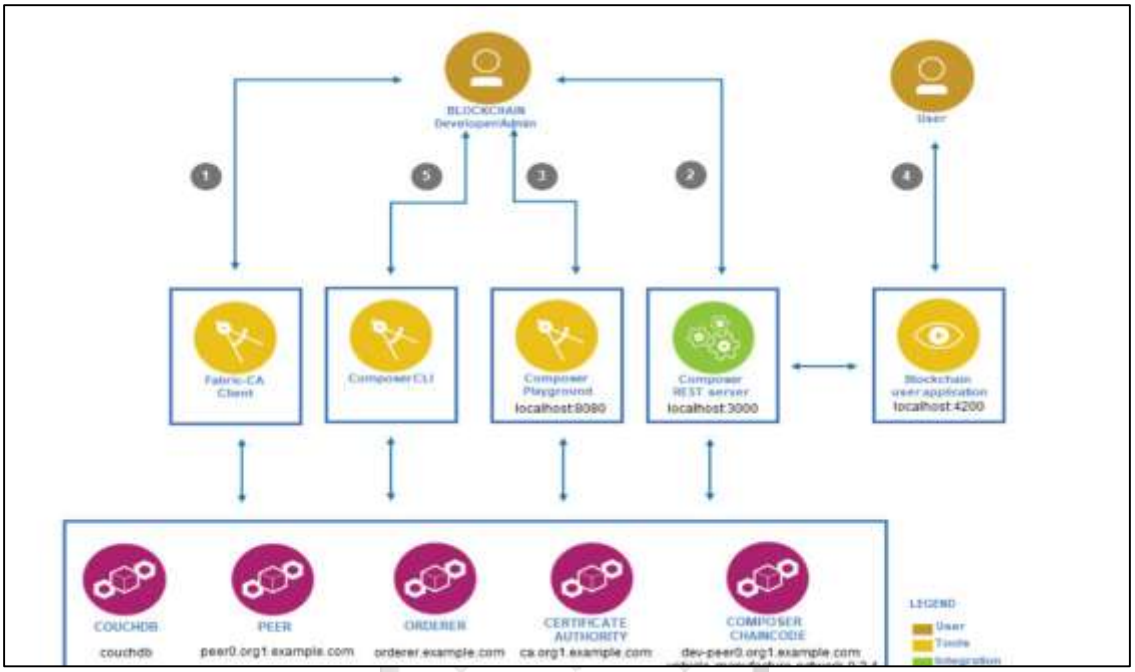


Figure 4.11: High level System Architecture

The Hyperledger Fabric components all run inside docker containers and are hosted on a single machine. The single peer for the deployed network uses 7051 and 7053 as request and event hub ports respectively. The CA listens on port 7054 while the single orderer listens on port 7050.

Using the Blockchain user application generated, student details such as Name, Course taken, Degree awarded, University name are added as an asset to the Hyperledger Fabric peer node by the university faculty administrator. A Record degree transaction is then issued by the faculty administrator to publish the transaction on the blockchain network and transfer ownership of the certificate to the respective student who will be able to share it with interested parties such as employers. To verify authenticity of provided academic certificates, a prospective employer or University enters the certificate identification number. The request is passed to the blockchain network over an Internet connection and user certificate details displayed on the web browser of the blockchain application. If the certificate details do not exist, an error message is displayed on the application graphical user interface.

CHAPTER FIVE

SYSTEM IMPLEMENTATION AND TESTING

5.1 Introduction

This section describes the various implementation and testing phases of underwent during the development of the system. The implementation illustrates the different hardware and software platforms used to develop the system. The user interfaces of the system are also described here.

5.2 System Development Environment

Various technologies made it possible to develop the system. These include the use of cloud computing technologies to host the development computer and virtualization techniques that allowed different blockchain services to run simultaneously without interfering with each other.

5.2.1 Kenya Education Network (KENET) Virtual Computing lab

KENET has developed a cloud-based platform that allows staff, faculty and students affiliated with KENET member institutions to access compute resources at no cost for a limited period of time. Verified users can provision virtual machine preloaded with different linux images in just a few seconds on their browsers. The resources are accessed at https://vlab.ac.ke.

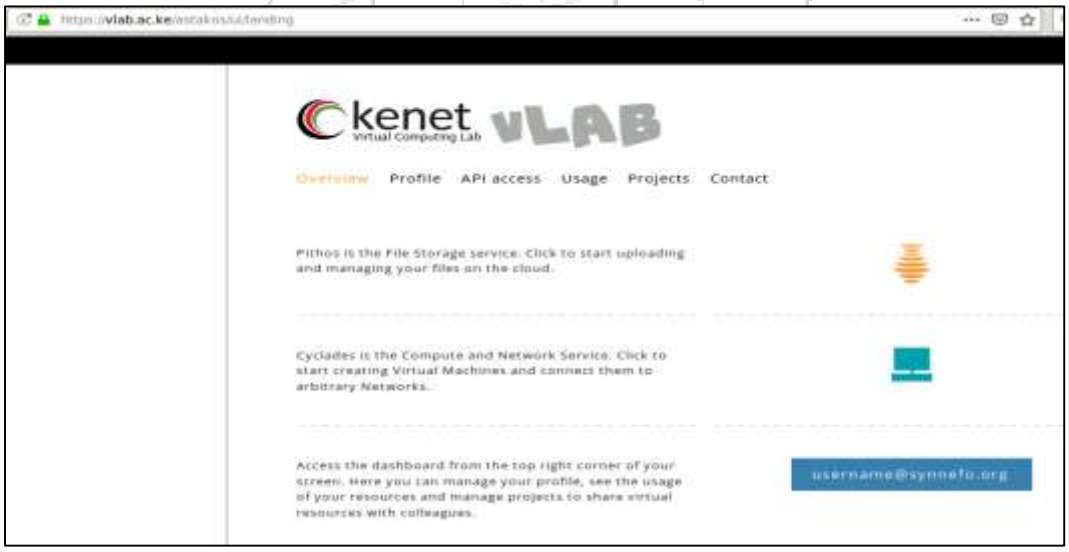


Figure 5.1: KENET Virtual Lab

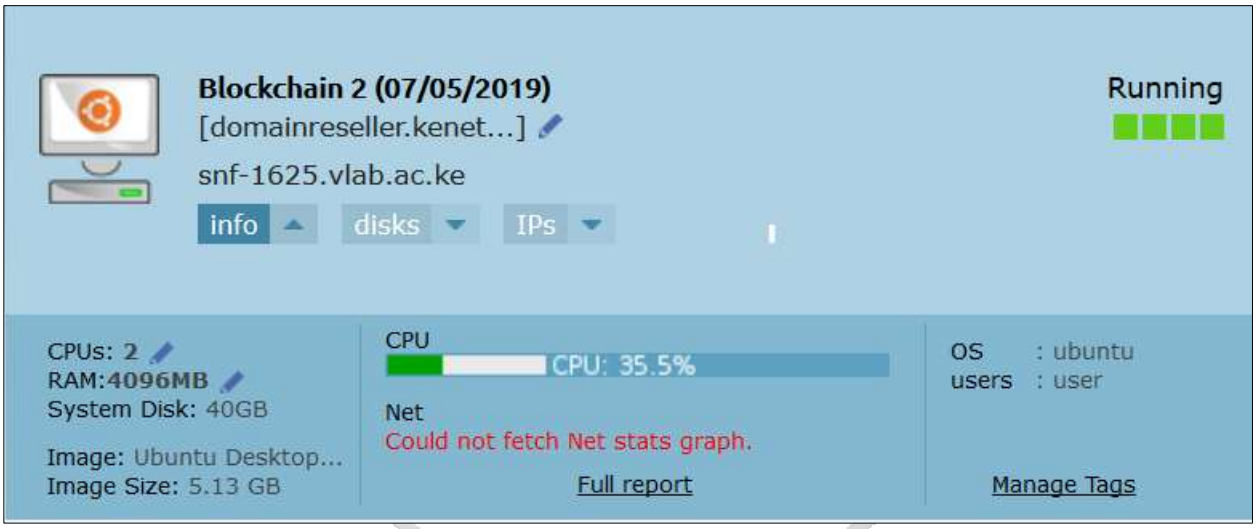


Figure 5.2: Resources of the Virtual Machine Provisioned on the KENET Vlab Platform

This cloud computing platform was chosen as the hosting solution due to the ease of use and the zero costs associated since Strathmore University is a KENET member.

The minimum requirement for the blockchain development platform was a server with at least 4GB of RAM. An Ubuntu 16.04 Virtual machine with 2VCPUs and 4GB of Ram was provisioned on the virtual computing platform and later installed with the Hyperledger fabric and composer frameworks alongside their prerequisites. The development environment was local host based and made use of a single user to demonstrate the functionality of the system.

5.2.2 Blockchain Application Prototype

The Hyperledger Composer supports creation of web, mobile or node.js applications. The composer-rest-server which is based on loopback technology was used to generate a REST API for the business network and the Hyperledger-composer Yaemon code generator was used to generate the angular application. The research made use of Blockchain4openscience degree business network was developed using the composer modelling language and JavaScript.

5.3 System Functionality Summary

The blockchain business network in use is designed to enable University faculty administrators to be able to generate academic certificates for individuals which are later published and stored on the blockchain network. The employers can then enter a student certificate identification number in order to know the legitimacy of the academic certificate presented.

5.4 Fundamental System User Interfaces

5.4.1 Starting Hyperledger fabric

The Hyperledger Fabric is started by executing the startFabric.sh script.

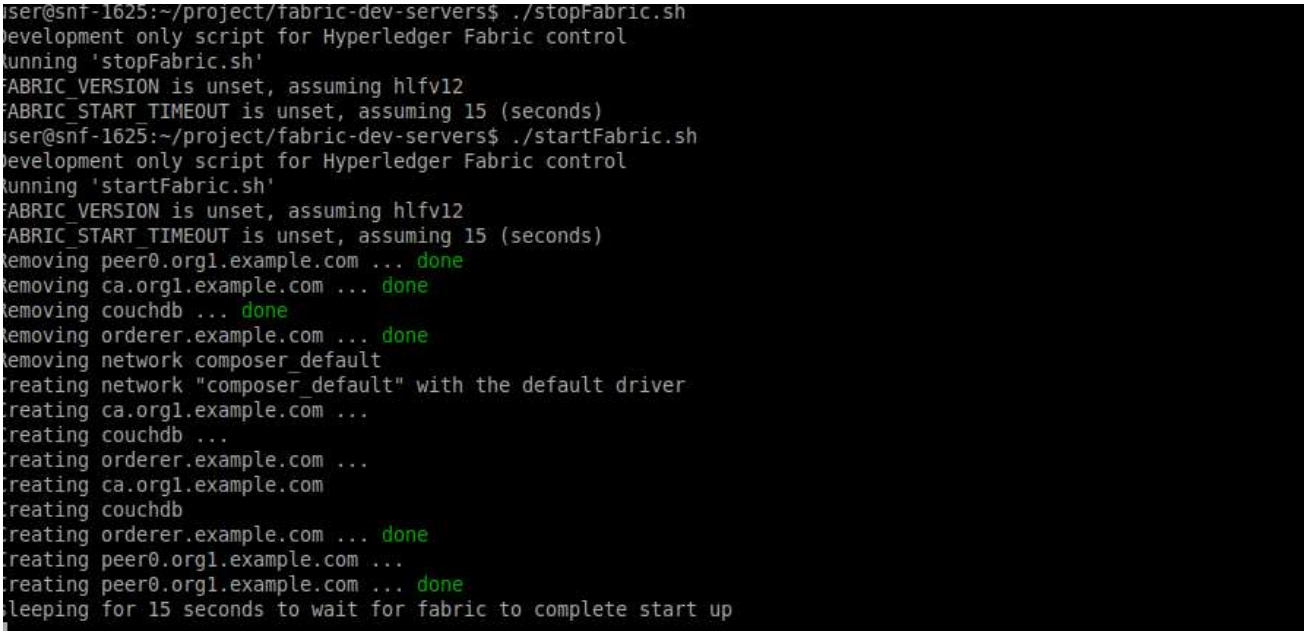


Figure 5.2: Hyperledger Fabric Command Line Startup

5.4.2 Starting Hyperledger Composer Rest Server

The Composer Rest Server is responsible for Generating the Rest API and is started by running the command shown in Figure 5.4.

```
user@snf-1625:~/academic-certificate-project$ composer-rest-server -c admin@academic-certificate-project -n always -w true
Discovering types from business network definition ...
Discovering the Returning Transactions..
Discovered types from business network definition
Generating schemas for all types in business network definition ...
Generated schemas for all types in business network definition
Adding schemas for all types to Loopback ...
Added schemas for all types to Loopback
Web server listening at: http://localhost:3000
Browse your REST API at http://localhost:3000/explorer
```

Figure 5.3: Hyperledger Composer Rest Server Startup

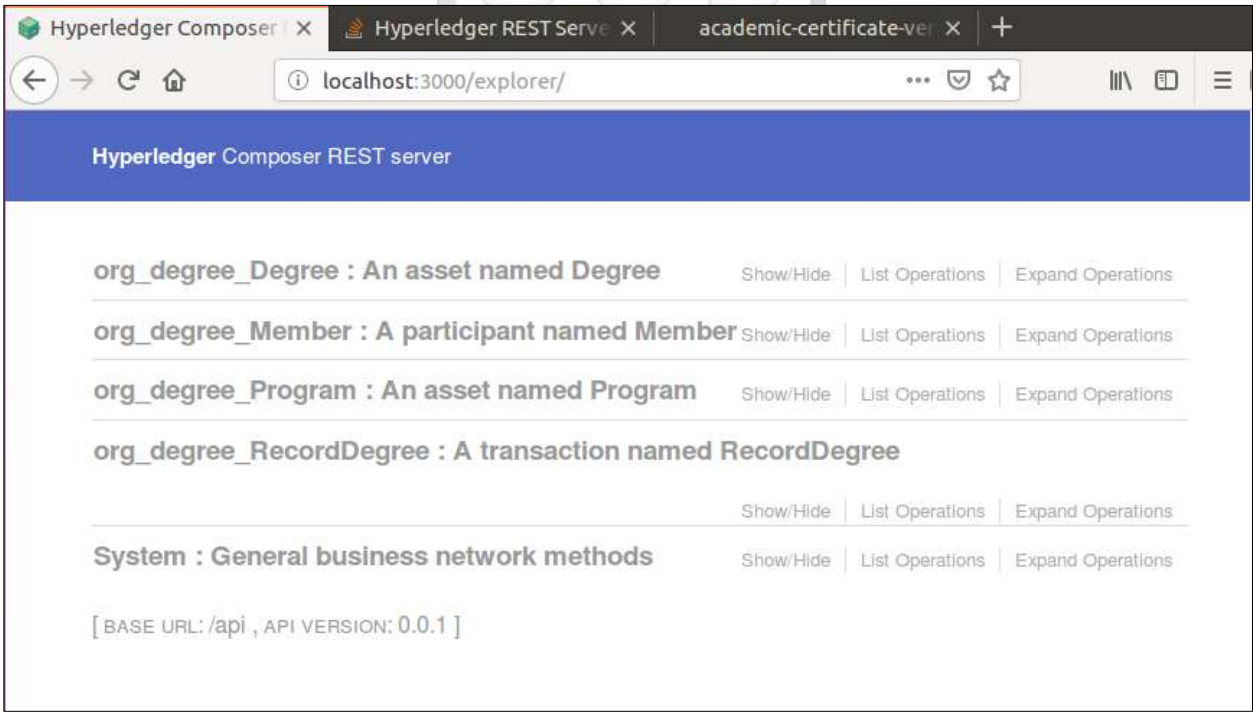


Figure 5.5: Hyperledger Composer Rest Server Web Page

5.4.3 Starting the Angular Blockchain Application

Once the Rest API is generated, the angular application is generated as shown in Figure 5.6.

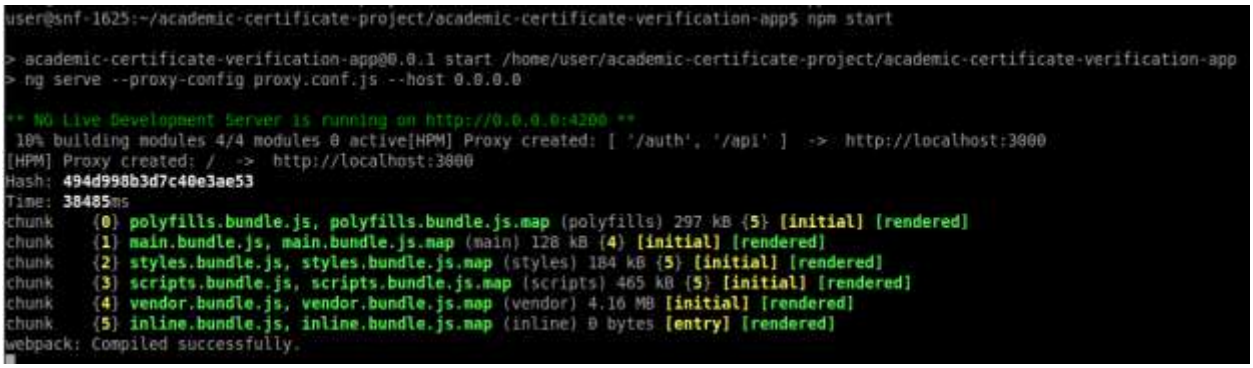


Figure 5.6: Angular Application Startup Process

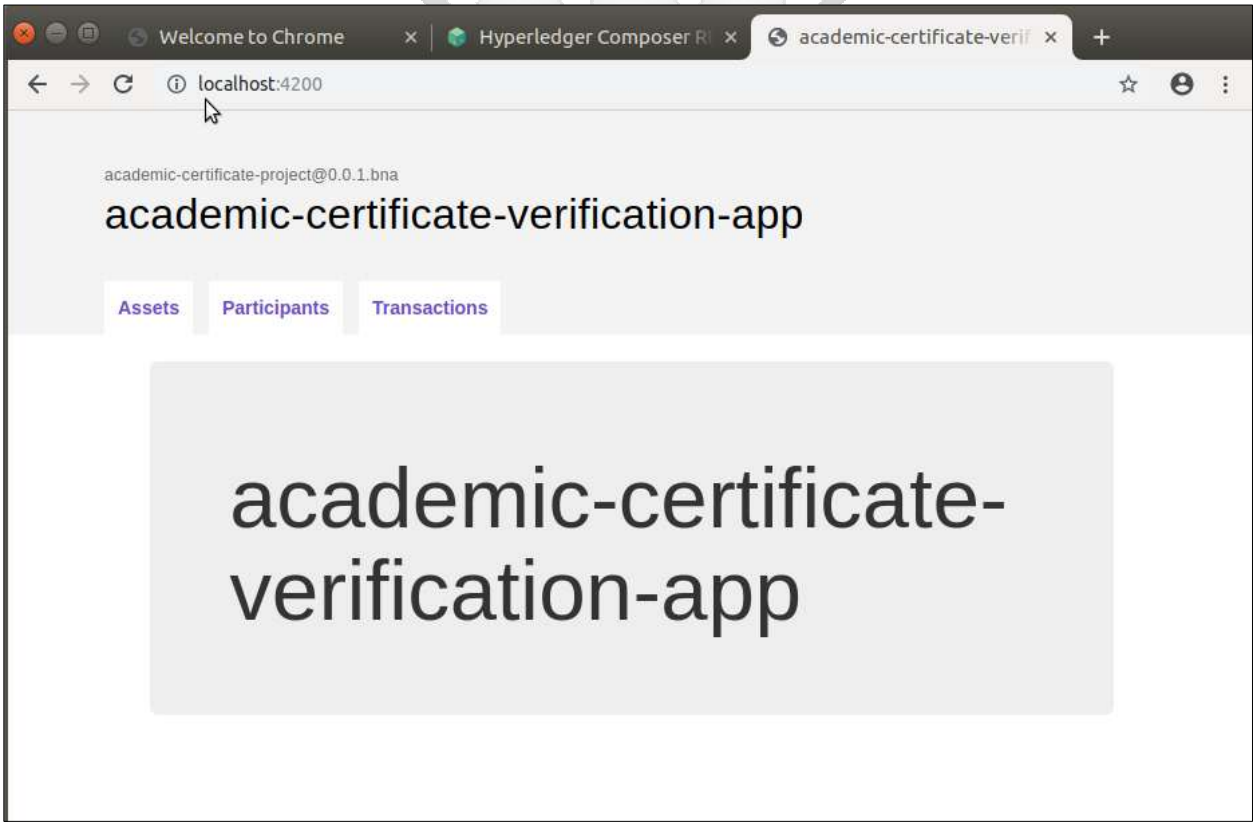


Figure 5.7: Certificate Verification Application Web Page

5.4.4 Adding Member Participants for the Business Network

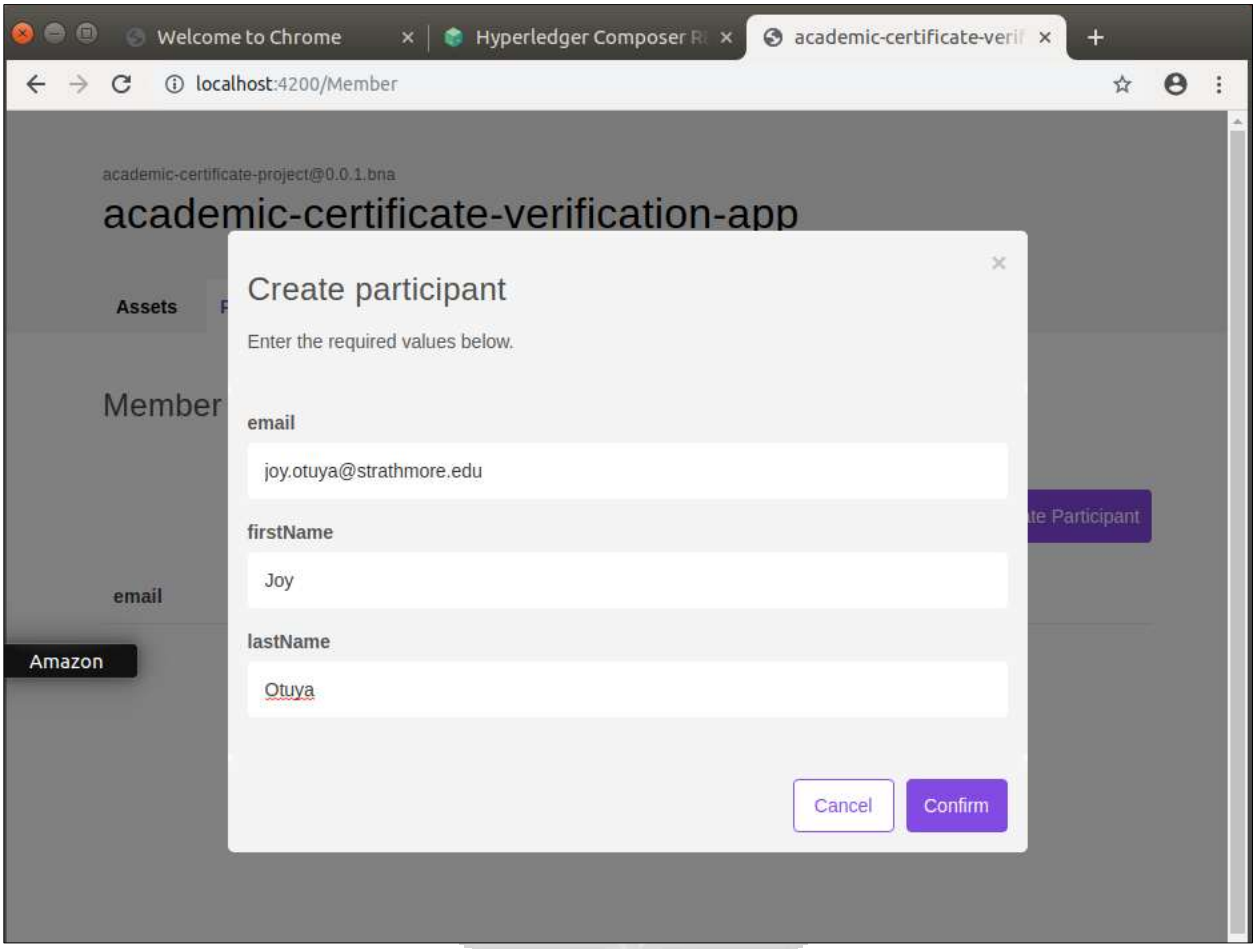


Figure 5.8: Adding Members to the Business Network

Figure 5.9 shows two members added one representing a student and the other the faculty administrator.

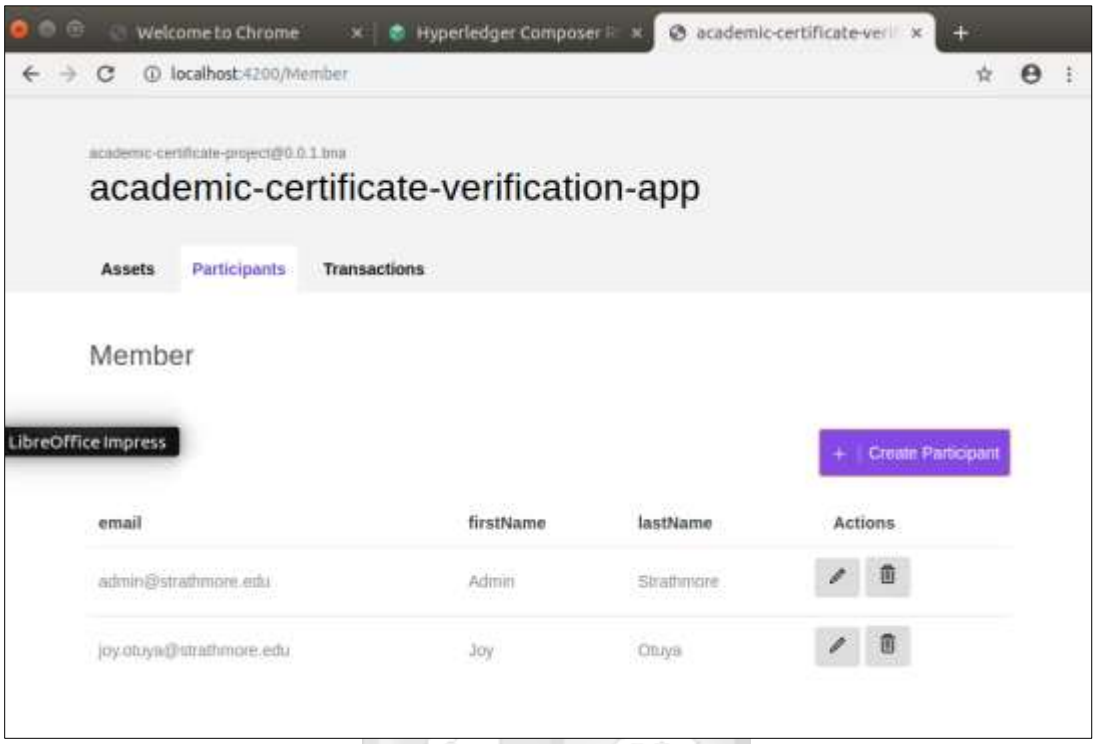


Figure 5.9: Members Added to the Business Network

5.4.5 Adding Program and Degree Assets for the Business Network

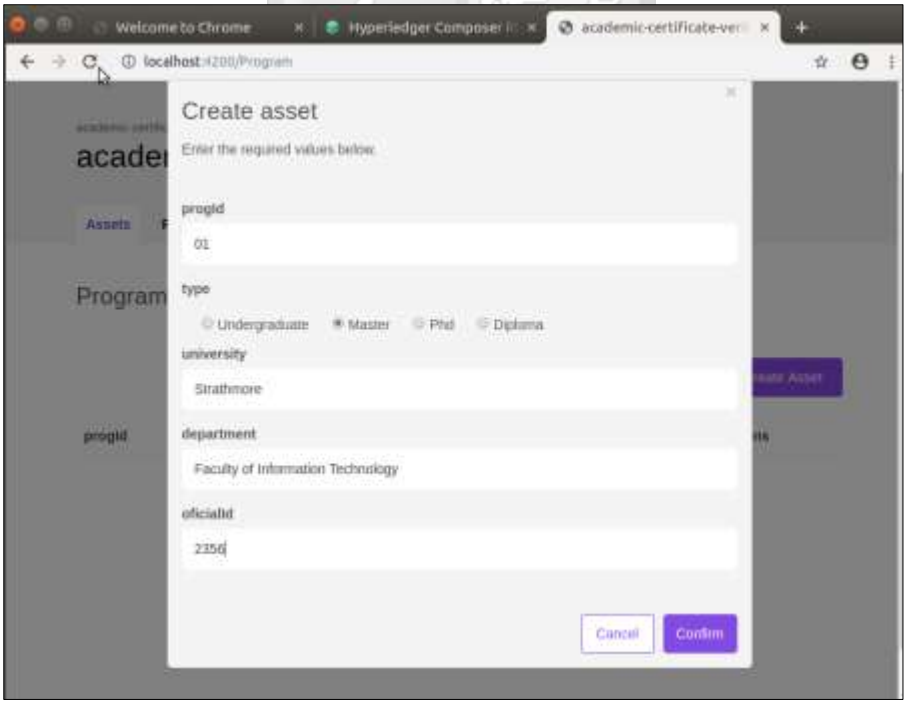


Figure 5.10: Adding a Faculty Program to the Business Network

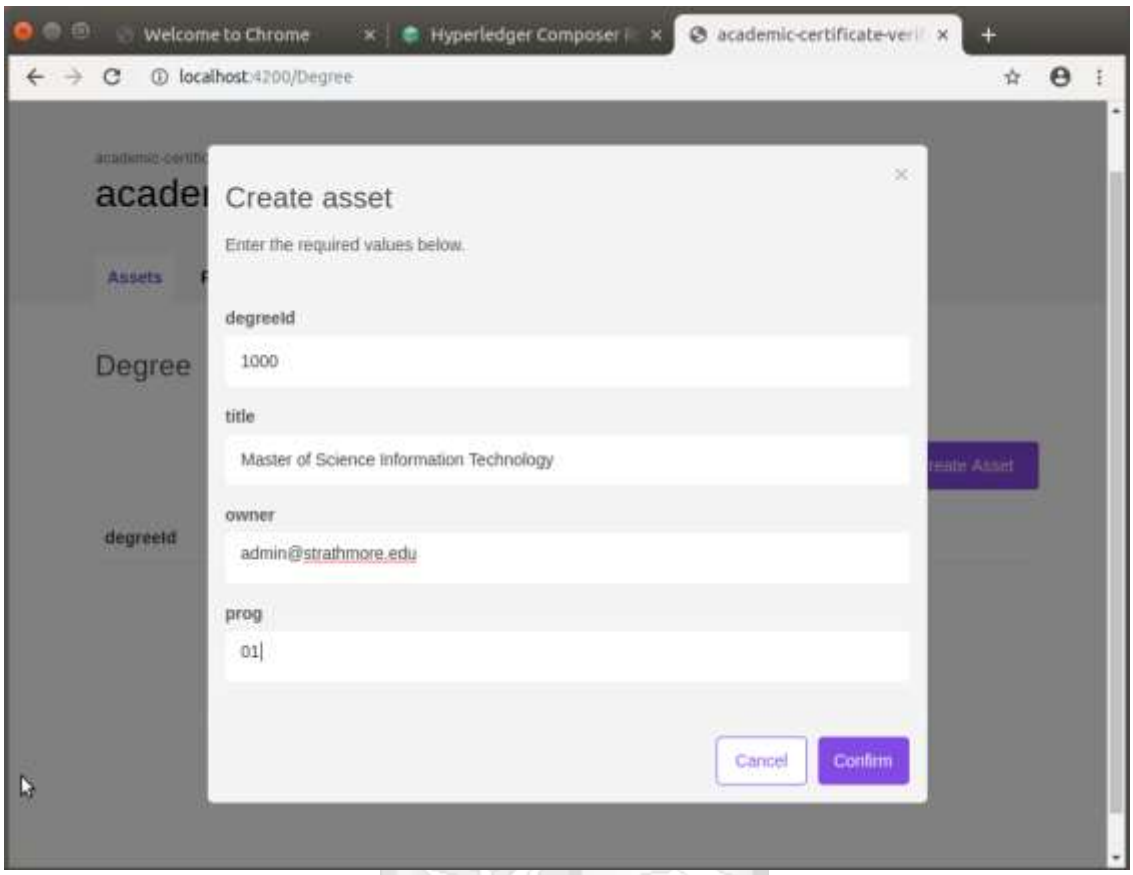


Figure 5.11: Adding a Faculty Degree to the Business Network

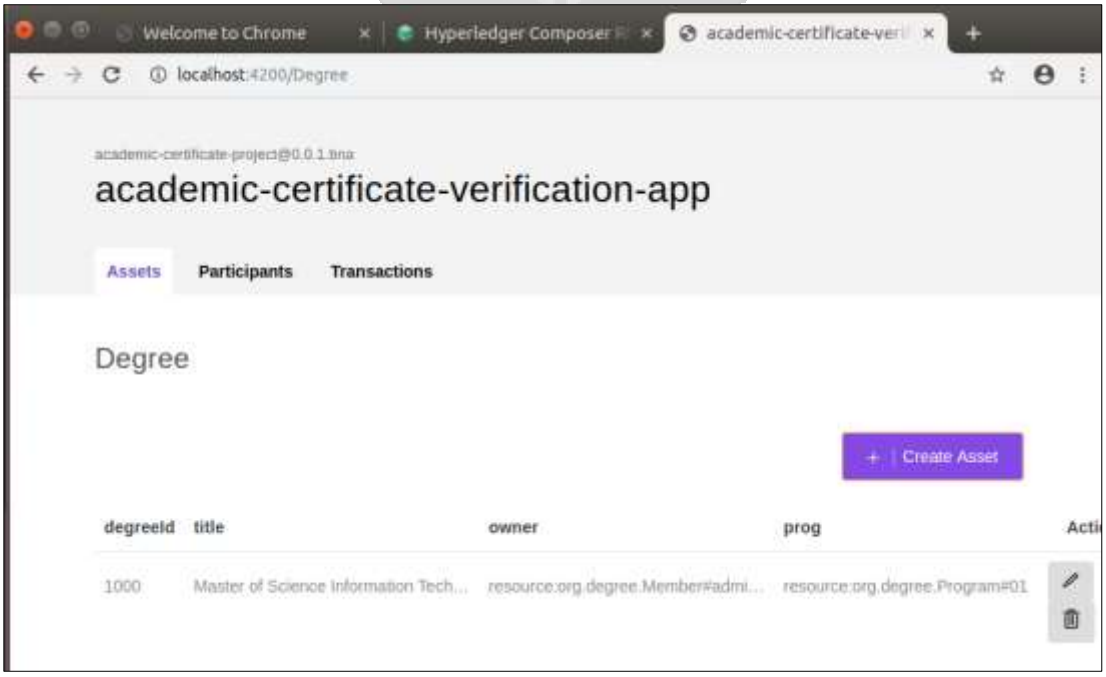


Figure 5.12: Degree Added to the Business Network

Figure 5.12 shows a degree details added to the ledger and has its owner as the faculty administrator.

5.4.6 Asset Ownership Transfer

Figure 5.13 and Figure 5.14 shows the user interfaces for invoking the transaction to transfer ownership of a degree from the faculty administrator to a student.

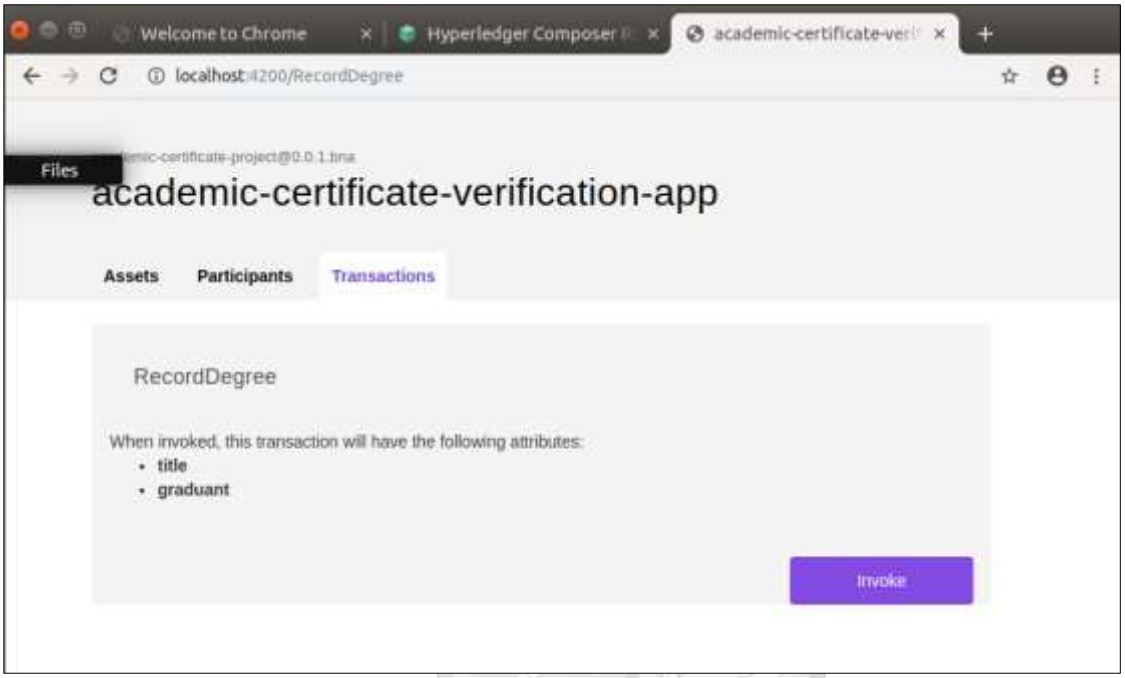


Figure 5.13: Invoke the Record Degree Transaction

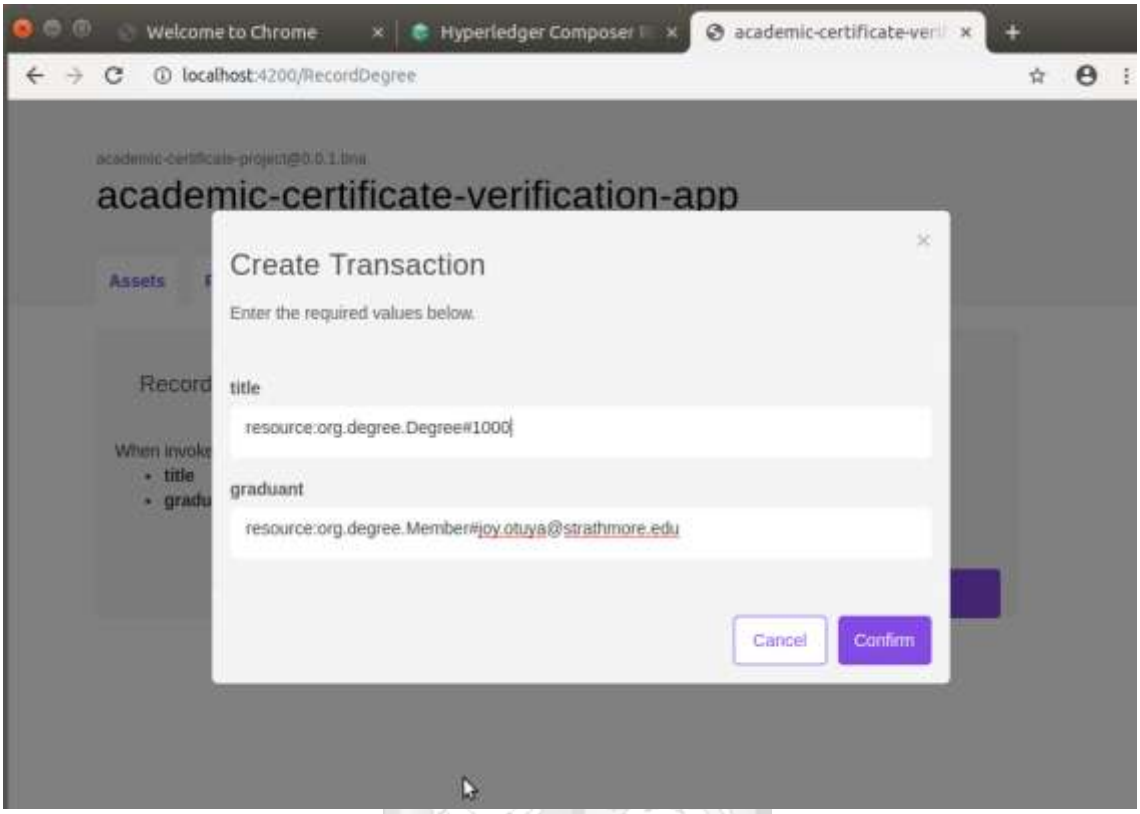


Figure 5.14: Enter Transaction Details

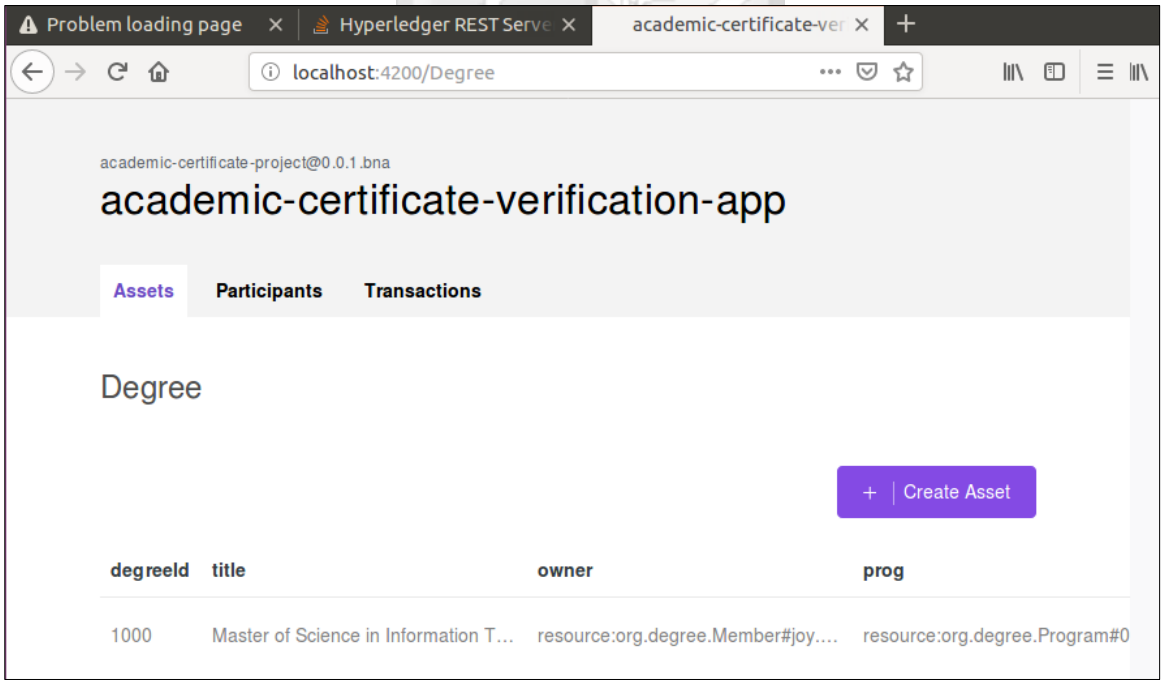


Figure 5.15: Asset Degree Changed Ownership to Student

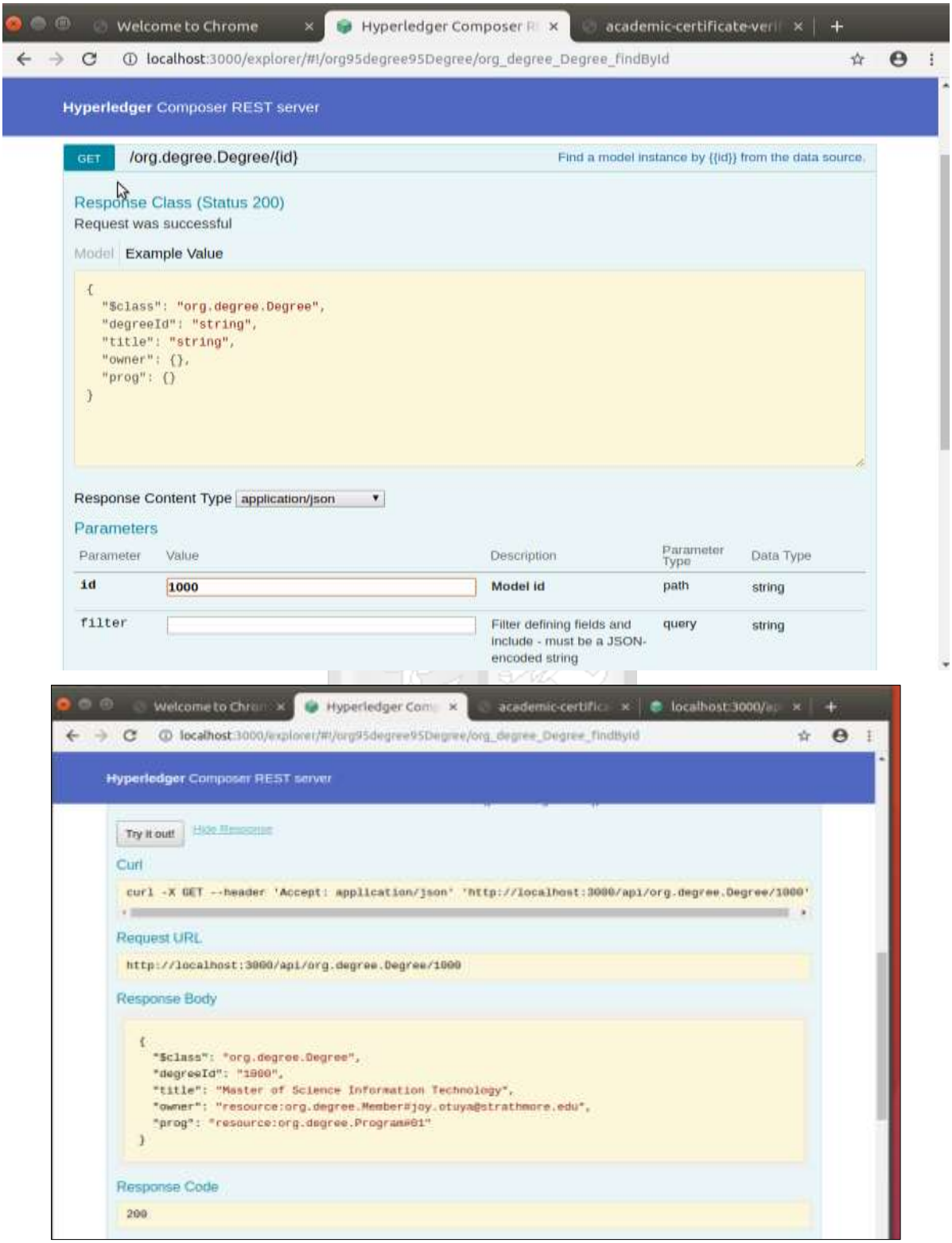


Figure 5.16: Confirming a Student's Certificate Details by Entering ID Certificate Details

To verify a certificate, the student's certificate identification number is fed into the system. A response code of 200 indicates that the certificate details exist in the ledger ad hence the certificate presented is deemed as valid. Else it is invalid.

5.5 System Testing

System testing was carried out to evaluate the systems compliance with the functional requirements specified earlier. The blockchain application was subjected to functional and compatibility tests as described in section 5.5.1 and 5.5.2. Integration testing was also carried out.

5.5.1 Functional Testing

Table 5.1: Summary of the Functional Tests Conducted

No	Test	Expected Results	Achieved Results
1	Loading of Blockchain Application	The system to be successfully launched via a web browser	The system was successfully launched on a web browser
2	Connecting to the Blockchain Business Network	A user with valid credentials should be allowed to connect to the blockchain business network	An authorised user was able to successfully connect to the blockchain business network
3	Generating user certificates	An authenticated administrator should be able to successfully generate user certificates.	An authenticated user was able to successfully generate certificates
4	Verification of issued certificates	An employer should be able to verify a user's certificate by entering certificate Identification Number	An employer was able to successfully verify a user's certificate.

5.5.2 Compatibility Testing

The application was developed and hosted on Ubuntu a Linux based Operating system. Test were later carried out to determine how the application launched on the latest versions of different web browsers.

Table 5.2: Compatibility Test Outcomes

Browser Type	Compatibility
Internet Explorer	Yes
Google Chrome	Yes
Mozilla Firefox	Yes

5.5.3 Integration Testing

This was done with an aim of confirming whether the Hyperledger composer and Hyperledger fabric integrated seamlessly.

CHAPTER SIX

DISCUSSION

6.1 Introduction

This research was aimed at analysing the different techniques and models used for detecting academic certificate forgery, analysing the blockchain technology and its applications, examining different theories and frameworks that were available for developing blockchain applications and finally to design, develop and test a blockchain application to counter forgery of university certificates in Kenya. This section therefore discusses the achievements attained in relation to the research objectives.

6.2 Review of the Research Objectives for the Blockchain Application

Through reviewing different literature sources, the first, second and third objectives were achieved successfully.

The first and second objective were to identify challenges of academic certificate verification and to analyse the different techniques and models used for detecting education certificate forgery respectively. It was noted that the main challenge was on manual verification methods which were expensive and time consuming. Literature revealed several techniques that are currently used for verification of academic certificates. These include manual verification, use of QR codes, web-based certificate verification and cryptographic techniques such as the use of ECDSA. It was noted that the existing certificate verification system had challenge such as time consumption and tediousness for the manual process and lack of assurance of records immutability. The blockchain solution was therefore seen as the most appropriate technology to use as it offers provenance, immutability with strong cryptographic mechanisms in place.

The third objective was to review existing frameworks for development of blockchain applications. This was to enable the researcher to know the existing blockchain development framework to choose the most appropriate one. The Hyperledger platform was chosen as the most appropriate platform because of familiarity and ease of use.

The architecture of the blockchain technology together with the building block components were also discussed. This was meant to help the researcher have a clear understanding of the blockchain technology. The various applications of the blockchain technology were also identified.

The fourth and final objective was to design, develop and test a blockchain application to counter forgery of university certificates in Kenya. This was achieved through actual design, implementation and testing of the blockchain solution. Use cases, sequence diagrams and data flow diagrams aided in the design phase of the application. Hyperledger Fabric and Composer were the main frameworks used to develop the application. The responsiveness and functionality of the system was tested on different operating systems and browsers.

6.3 System Assessment

The blockchain application prototype was developed and requires a user to have internet connectivity to access the systems services. The following section briefly describes the advantages and disadvantages presented by the application.

6.3.1 Advantages of the Developed Blockchain Solution

i. The application can be used across all platforms i.e operating systems and browsers.
ii. The disadvantages of time wastage and errors introduced by the manual verification process are eliminated.
iii. Enhanced security of records as compared to other systems due to strong cryptographic mechanisms in place.
iv. With the blockchain application, immutability and provenance are realised.

6.3.2 Disadvantages of the Developed Blockchain Solution

i. The application does not provide an option for reporting fake certificate.
ii. The blockchain application requires a user to have internet connectivity to use the system.
iii. The application does not address issues that may arise from impersonation.
iv. The application is only limited to the verification of university certificates.

CHAPTER SEVEN

CONCLUSION, RECOMMENDATIONS AND FUTURE WORK

7.1 Conclusion

The study was aimed at developing a blockchain application to counter forgery of university certificates by publicly providing evidence that a student received a certificate from a certain university in Kenya. This was made possible by reviewing existing literature to examine the different methods used for academic certificate verification, analysing the blockchain technology architecture and finally identification of a suitable framework to use for the development of the application.

A prototype was developed that allowed a user's certificate to be generated, system participants to be added and finally the verification of a student's certificate by an external party such as an employer. The system was then tested to ensure that it was able to meet the functional requirements.

7.2 Recommendations

This study has presented a relatively new solution to address the issue of academic certificate falsification in Kenya. However, for universities to reap maximum benefit from such technology, the following is recommended:

i. The universities are encouraged to adopt automated methods of certificate verification by striving to develop a common blockchain network where academic certificate records for the participating universities can be stored in a common ledger.

7.3 Future Work

The following are the recommendations for future work relating to the blockchain application

i. The application can be scaled to tie up the whole academic achievement records of a university such as transcript tracking.

ii. The application can be further developed to integrate with an external web-based system that has a more user-friendly graphical interface.
iii. The application be developed further to include visual certificate components such as signatures, logos and seals. A passport phot section could also be introduced to counter impersonation cases.
iv. The application can be scaled further and incorporate external storage of the digital certificates instead of storing them on the blockchain. Such that the blockchain will only store pointers to objects on the external storage. This is the most appropriate way of storing the records.

REFERENCES

Apte, S., & Petrovsky, N. (2016). Will blockchain technology revolutionize excipient supply chain management? *Journal of Excipients and Food Chemicals*, *7*(3), 910.

Asadi, A., Rahbar, N., Rezvani, M. J., & Asadi, F. (2018). Fake/bogus conferences: Their features and some subtle ways to differentiate them from real ones. *Science and Engineering Ethics*, *24*(2), 779–784.

Burge, D. S. (2011). *The Systems Engineering Tool Box*. 24.

Decoo, W. (2002). *Crisis on campus*. The MIT Press.

Feiler, P., & S. Humphrey, W. (1993, March 25). *Software Process Development and Enactment: Concepts and Definitions*. 28–40. https://doi.org/10.1109/SPCON.1993.236824

Garwe, E. (2014). Quality assurance in higher education in Zimbabwe. *Research in Higher Education Journal*, *23*.

Golosova, J., & Romanovs, A. (2018, November 1). *The Advantages and Disadvantages of the Blockchain Technology*. 1–6. https://doi.org/10.1109/AIEEE.2018.8592253

Hyperledger. (2018). Hyperledger – Open Source Blockchain Technologies. Retrieved December 4, 2018, from Hyperledger website: https://www.hyperledger.org/

Jeppsson, A., & Olsson, O. (2017). Blockchains as a solution for traceability and transparency. Retrieved from http://lup.lub.lu.se/student-papers/record/8919957

Jimu, I. M. (2018). Fake Qualifications and the Challenge of regulating Higher Education in Southern Africa. *Modern Africa: Politics, History and Society*, *6*(1), 107–134. https://doi.org/10.26806/modafr.v6i1.236

Kaibiru, R. M., & Shibwabo, B. (2017). A Prototype for authentication of secondary school certificates: a case of Kenya certificate of secondary education.

KNQA_ICT. (2018). Verify Your Qualifications in Kenya. Retrieved June 10, 2019, from Kenya National Qualifications Authority website: http://www.knqa.go.ke/verify-your-qualifications-in- kenya/

Kosba, A., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016). Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE symposium on security and privacy (SP)* (pp. 839–858). IEEE.

Lemieux, V. L. (2016). Trusting records: is Blockchain technology the answer? *Records Management Journal*, *26*(2), 110–139.

Maqhina, M. (2018). Shock increase in fake credentials | IOL News. Retrieved June 7, 2019, from https://www.iol.co.za/news/politics/shock-increase-in-fake-credentials-18264458

Malkawi, M. I. (2017). Counterfeit Prevention and Detection of University and Academic Institutions Documents Using Unique Codes.

Morris, V., Adivi, R., Asara, R., Cousens, M., Gupta, N., Lincoln, N., … Sun, H. W. (2018, May 30). Developing a Blockchain Business Network with Hyperledger Composer using the IBM Blockchain Platform Starter Plan. Retrieved 4 December 2018, from http://www.redbooks.ibm.com/abstracts/redp5492.html?Open

Murthy, S., Murthy, R. M., & Sarma, A. C. (2011). Elliptic Curve based Signature Method to Control Fake Paper based Certificates. In *Proceedings of the World Congress on Engineering and Computer Science* (Vol. 1, pp. 19–21).

New York Times. (2015). Opinion | A Rising Tide of Bogus Degrees - The New York Times. Retrieved June 7, 2019, from https://www.nytimes.com/2015/05/20/opinion/a-rising-tide-of-bogus- degrees.html

Ochieng, H. O. (2016). *A mobile based application for verification of legitimacy of degree certificates in Kenya* (Thesis, Strathmore University). Retrieved from http://su-plus.strathmore.edu/handle/11071/4904

Pilkington, M. (2016). 11 Blockchain technology: principles and applications. *Research Handbook on Digital Transformations*, 225.

Sharples, M., & Domingue, J. (2016). The Blockchain and Kudos: A Distributed System for Educational Record, Reputation and Reward. In K. Verbert, M. Sharples, & T. Klobučar (Eds.), *Adaptive and Adaptable Learning* (pp. 490–496). Springer International Publishing.

Solomon, N. (2016). Mr. President, fake academic certificates are choking Kenya's economy and job market. Retrieved December 4, 2018, from UREPORT-CITIZEN JOURNALISM

website: https://www.standardmedia.co.ke/ureport/story/2000193042/mr-president-fake-academic- certificates-are-choking-kenya-s-economy-and-job-market

Some, K. (2017, July 23). Judiciary probes staff 'fake papers'. Retrieved 4 December 2018, from https://www.nation.co.ke/news/judiciary-staff-under-probe-over-fake-papers/1056-4027994-x9gr87/index.html

Tongkaw, S., Inkaew, W., & Tongkaw, A. (2019, February 4). *RAD Design and Data Management Systems of Natural Resources and Local Wisdom*.

Turkanović, M., Hölbl, M., Košič, K., Heričko, M., & Kamišalić, A. (2018). EduCTX: A Blockchain-Based Higher Education Credit Platform. *IEEE Access*, *6*, 5112–5127. https://doi.org/10.1109/ACCESS.2018.2789929

Underwood, S. (2016). Blockchain beyond bitcoin. *Communications of the ACM*, *59*(11), 15–17.

Van Tol, J. E. (1990). Detecting, deterring and punishing the use of fraudulent academic credentials: A play in two acts. *Santa Clara L. Rev.*, *30*, 791.

Wood, D. G. (2014). *ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER*. 32.

Yusuf, S. I., Boukar, M. M., & Muslu, I. (2017). Data dissemination via web services for distributed and heterogeneous data sources: An enhancement of the nigerian university certificate verification system. In *2017 13th International Conference on Electronics, Computer and Computation (ICECCO)* (pp. 1–4). https://doi.org/10.1109/ICECCO.2017.8333323

Zahid, J., Hussain, F., & Ferworn, A. (2016). Integrating Internet of Things and blockchain: use cases. *Newsletter, 2016*.

Zastrin, I. (2019). What is Ethereum. Retrieved June 1, 2019, from https://www.zastrin.com/courses/ethereum-primer/lessons/1-1

Zhou, Y., & Chow, S. T. (2009). System and method of hiding cryptographic private keys.

Zignuts Technolab, A. (2018, May 23). How blockchain architecture works? Basic Understanding of Blockchain and its Architecture. Retrieved 4 December 2018, from https://www.zignuts.com/blogs/how-blockchain-architecture-works-basic-understanding-of-blockchain-and-its-architecture/

APPENDIX A: Background processes when starting the Hyperledger Fabric

2019-04-19 08:45:34.788 UTC [viperutil] getKeysRecursively -> DEBU 001 Found map[string]interface{} value for peer.BCCSP

2019-04-19 08:45:34.788 UTC [viperutil] getKeysRecursively -> DEBU 002 Found map[string]interface{} value for peer.BCCSP.PKCS11

2019-04-19 08:45:34.788 UTC [viperutil] unmarshalJSON -> DEBU 003 Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:34.789 UTC [viperutil] getKeysRecursively -> DEBU 004 Found real value for peer.BCCSP.PKCS11.Hash setting to <nil> <nil>

2019-04-19 08:45:34.789 UTC [viperutil] unmarshalJSON -> DEBU 005 Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:34.790 UTC [viperutil] getKeysRecursively -> DEBU 006 Found real value for peer.BCCSP.PKCS11.Security setting to <nil> <nil>

2019-04-19 08:45:34.791 UTC [viperutil] getKeysRecursively -> DEBU 007 Found map[string]interface{} value for peer.BCCSP.PKCS11.FileKeyStore

2019-04-19 08:45:34.791 UTC [viperutil] unmarshalJSON -> DEBU 008 Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:34.792 UTC [viperutil] getKeysRecursively -> DEBU 009 Found real value for peer.BCCSP.PKCS11.FileKeyStore.KeyStore setting to <nil> <nil>

2019-04-19 08:45:34.792 UTC [viperutil] unmarshalJSON -> DEBU 00a Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:34.793 UTC [viperutil] getKeysRecursively -> DEBU 00b Found real value for peer.BCCSP.PKCS11.Library setting to <nil> <nil>

2019-04-19 08:45:34.794 UTC [viperutil] unmarshalJSON -> DEBU 00c Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:34.794 UTC [viperutil] getKeysRecursively -> DEBU 00d Found real value for peer.BCCSP.PKCS11.Label setting to <nil> <nil>

2019-04-19 08:45:34.794 UTC [viperutil] unmarshalJSON -> DEBU 00e Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:34.795 UTC [viperutil] getKeysRecursively -> DEBU 00f Found real value for peer.BCCSP.PKCS11.Pin setting to <nil> <nil>

2019-04-19 08:45:34.796 UTC [viperutil] unmarshalJSON -> DEBU 010 Unmarshal JSON: value cannot be unmarshalled: invalid character 'S' looking for beginning of value

2019-04-19 08:45:34.796 UTC [viperutil] getKeysRecursively -> DEBU 011 Found real value for peer.BCCSP.Default setting to string SW

2019-04-19 08:45:34.796 UTC [viperutil] getKeysRecursively -> DEBU 012 Found map[string]interface{} value for peer.BCCSP.SW

2019-04-19 08:45:34.797 UTC [viperutil] getKeysRecursively -> DEBU 013 Found map[string]interface{} value for peer.BCCSP.SW.FileKeyStore

2019-04-19 08:45:34.797 UTC [viperutil] unmarshalJSON -> DEBU 014 Unmarshal JSON: value cannot be unmarshalled: unexpected end of JSON input

2019-04-19 08:45:34.797 UTC [viperutil] getKeysRecursively -> DEBU 015 Found real value for peer.BCCSP.SW.FileKeyStore.KeyStore setting to string

2019-04-19 08:45:34.797 UTC [viperutil] unmarshalJSON -> DEBU 016 Unmarshal JSON: value cannot be unmarshalled: invalid character 'S' looking for beginning of value

2019-04-19 08:45:34.798 UTC [viperutil] getKeysRecursively -> DEBU 017 Found real value for peer.BCCSP.SW.Hash setting to string SHA2

2019-04-19 08:45:34.798 UTC [viperutil] unmarshalJSON -> DEBU 018 Unmarshal JSON: value is not a string: 256

2019-04-19 08:45:34.798 UTC [viperutil] getKeysRecursively -> DEBU 019 Found real value for peer.BCCSP.SW.Security setting to int 256

2019-04-19 08:45:34.798 UTC [viperutil] EnhancedExactUnmarshalKey -> DEBU 01a map[peer.BCCSP:map[PKCS11:map[Label:<nil> Pin:<nil> Hash:<nil> Security:<nil> FileKeyStore:map[KeyStore:<nil>] Library:<nil>] Default:SW SW:map[FileKeyStore:map[KeyStore:] Hash:SHA2 Security:256]]]

2019-04-19 08:45:34.799 UTC [bccsp_sw] openKeyStore -> DEBU 01b KeyStore opened at [/etc/Hyperledger/peer/msp/keystore]...done

2019-04-19 08:45:34.800 UTC [bccsp] initBCCSP -> DEBU 01c Initialize BCCSP [SW]

2019-04-19 08:45:34.800 UTC [msp] getPemMaterialFromDir -> DEBU 01d Reading directory /etc/Hyperledger/peer/msp/signcerts

2019-04-19 08:45:34.800 UTC [msp] getPemMaterialFromDir -> DEBU 01e Inspecting file /etc/Hyperledger/peer/msp/signcerts/peer0.org1.example.com-cert.pem

2019-04-19 08:45:34.800 UTC [msp] getPemMaterialFromDir -> DEBU 01f Reading directory /etc/Hyperledger/peer/msp/cacerts

2019-04-19 08:45:34.800 UTC [msp] getPemMaterialFromDir -> DEBU 020 Inspecting file /etc/Hyperledger/peer/msp/cacerts/ca.org1.example.com-cert.pem

2019-04-19 08:45:34.800 UTC [msp] getPemMaterialFromDir -> DEBU 021 Reading directory /etc/Hyperledger/peer/msp/admincerts

2019-04-19 08:45:34.800 UTC [msp] getPemMaterialFromDir -> DEBU 022 Inspecting file /etc/Hyperledger/peer/msp/admincerts/Admin@org1.example.com-cert.pem

2019-04-19 08:45:34.801 UTC [msp] getPemMaterialFromDir -> DEBU 023 Reading directory /etc/Hyperledger/peer/msp/intermediatecerts

2019-04-19 08:45:34.801 UTC [msp] getMspConfig -> DEBU 024 Intermediate certs folder not found at [/etc/Hyperledger/peer/msp/intermediatecerts]. Skipping. [stat /etc/Hyperledger/peer/msp/intermediatecerts: no such file or directory]

2019-04-19 08:45:34.801 UTC [msp] getPemMaterialFromDir -> DEBU 025 Reading directory /etc/Hyperledger/peer/msp/tlscacerts

2019-04-19 08:45:34.801 UTC [msp] getPemMaterialFromDir -> DEBU 026 Inspecting file /etc/Hyperledger/peer/msp/tlscacerts/tlsca.org1.example.com-cert.pem

2019-04-19 08:45:34.801 UTC [msp] getPemMaterialFromDir -> DEBU 027 Reading directory /etc/Hyperledger/peer/msp/tlsintermediatecerts

2019-04-19 08:45:34.801 UTC [msp] getMspConfig -> DEBU 028 TLS intermediate certs folder not found at [/etc/Hyperledger/peer/msp/tlsintermediatecerts]. Skipping. [stat /etc/Hyperledger/peer/msp/tlsintermediatecerts: no such file or directory]

2019-04-19 08:45:34.801 UTC [msp] getPemMaterialFromDir -> DEBU 029 Reading directory /etc/Hyperledger/peer/msp/crls

2019-04-19 08:45:34.801 UTC [msp] getMspConfig -> DEBU 02a crls folder not found at [/etc/Hyperledger/peer/msp/crls]. Skipping. [stat /etc/Hyperledger/peer/msp/crls: no such file or directory]

2019-04-19 08:45:34.801 UTC [msp] getMspConfig -> DEBU 02b MSP configuration file not found at [/etc/Hyperledger/peer/msp/config.yaml]: [stat /etc/Hyperledger/peer/msp/config.yaml: no such file or directory]

2019-04-19 08:45:34.802 UTC [msp] newBccspMsp -> DEBU 02c Creating BCCSP-based MSP instance

2019-04-19 08:45:34.802 UTC [msp] New -> DEBU 02d Creating Cache-MSP instance

2019-04-19 08:45:34.802 UTC [msp] loadLocaMSP -> DEBU 02e Created new local MSP

2019-04-19 08:45:34.803 UTC [msp] Setup -> DEBU 02f Setting up MSP instance Org1MSP

2019-04-19 08:45:34.805 UTC [msp/identity] newIdentity -> DEBU 030 Creating identity instance for cert -----BEGIN CERTIFICATE-----

MIICQjCCAemgAwIBAgIQDJbOh88U+tlJ9He5sjUwBDAKBggqhkjOPQQDAjBzMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1UEBxMNU2F
uIEZy
YW5jaXNjbzEZMBcGA1UEChMQb3JnMS5leGFtcGxlLmNvbTEcMBoGA1UEAxMTY2Eu
b3JnMS5leGFtcGxlLmNvbTAeFw0xNzA2MjYxMjQ5MjZaFw0yNzA2MjQxMjQ5MjZa
MHMxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQ
HEw1T
YW4gRnJhbmNpc2NvMRkwFwYDVQQKExBvcmcxLmV4YW1wbGUuY29tMRwwGgYDV
QQD
ExNjYS5vcmcxLmV4YW1wbGUuY29tMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE
kQ216YBD8kks+IzKJyBmdLqh/L/sEZ5twTqYpsK1ztNhIUDplsletAFOJQWRH+kb
hzFFNvS1qwveGRU6ztN5haNfMF0wDgYDVR0PAQH/BAQDAgGmMA8GA1UdJQQIMAY
G
BFUdJQAwDwYDVR0TAQH/BAUwAwEB/zApBgNVHQ4EIgQgGatlq7sEgH2tEuTAqaqm
ZJ5who46vQIXoyLYnkfhpq4wCgYIKoZIzj0EAwIDRwAwRAIgCyrj/lUjtBYaEgMt
x98l5z+iLU6r+gp4CsdcDYzKLugCIGXlcU56avWSUtRAGn8Avpb6TOxtkrKIpeTE
QfM8VsS/
-----END CERTIFICATE-----

2019-04-19 08:45:34.805 UTC [msp/identity] newIdentity -> DEBU 031 Creating identity instance for cert -----BEGIN CERTIFICATE-----

MIICGjCCAcCgAwIBAgIRANuOnVN+yd/BGyoX7ioEklQwCgYIKoZIzj0EAwIwczEL

MAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNhbiB
G
cmFuY2lzY28xGTAXBgNVBAoTEG9yZzEuZXhhbXBsZS5jb20xHDAaBgNVBAMTE2Nh
Lm9yZzEuZXhhbXBsZS5jb20wHhcNMTcwNjI2MTI0OTI2WhcNMjcwNjI0MTI0OTI2
WjBbMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1U
EBxMN
U2FuIEZyYW5jaXNjbzEfMB0GA1UEAwwWQWRtaW5Ab3JnMS5leGFtcGxlLmNvbTBZ
MBMGByqGSM49AgEGCCqGSM49AwEHA0IABGu8KxBQ1GkxSTMVoLv7NXiYKWj5t6
Dh
WRTJBHnLkWV7lRUfYaKAKFadSii5M7Z7ZpwD8NS7IsMdPR6Z4EyGgwKjTTBLMA4G
A1UdDwEB/wQEAwIHgDAMBgNVHRMBAf8EAjAAMCsGA1UdIwQkMCKAIBmrZau7BI
B9
rRLkwKmqpmSecIaOOr0CF6Mi2J5H4aauMAoGCCqGSM49BAMCA0gAMEUCIQC4sKQ6
CEgqbTYe48az95W9/hnZ+7DI5eSnWUwV9vCd/gIgS5K6omNJydoFoEpaEIwM97uS
XVMHPa0iyC497vdNURA=
-----END CERTIFICATE-----
2019-04-19 08:45:34.867 UTC [msp/identity] newIdentity -> DEBU 032 Creating identity instance for cert -----BEGIN CERTIFICATE-----
MIICGTCCAb+gAwIBAgIQTx2TvwYtAf62KKQliP6UoTAKBggqhkjOPQQDAjBzMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1UEBxMNU2F
uIEZy
YW5jaXNjbzEZMBcGA1UEChMQb3JnMS5leGFtcGxlLmNvbTEcMBoGA1UEAxMTY2Eu
b3JnMS5leGFtcGxlLmNvbTAeFw0xNzA2MjYxMjQ5MjZaFw0yNzA2MjQxMjQ5MjZa
MFsxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQH
Ew1T
YW4gRnJhbmNpc2NvMR8wHQYDVQQDExZwZWVyMC5vcmcxLmV4YW1wbGUuY29tM
Fkw
EwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEPPHUp7+EYb2xIbleWfRClMgqbtQqRmIS
2a5F8T0L3J6IZp9wm7K+w4LIBIgw1Cz9D8nqHW6f4OYBrbp0cSGnR6NNMEswDgYD
VR0PAQH/BAQDAgeAMAwGA1UdEwEB/wQCMAAwKwYDVR0jBCQwIoAgGatlq7sEgH2
t

EuTAqaqmZJ5who46vQIXoyLYnkfhpq4wCgYIKoZIzj0EAwIDSAAwRQIhAK4i2Hz2
K398TvjJk62neDoenYhkMY7rBN3BN/GI0G0SAiAOTx36wuy9/4BBV8NVBCZ9V+Iw
msdI9CyZ59oVMVmNYQ==
-----END CERTIFICATE-----
2019-04-19 08:45:34.867 UTC [bccsp_sw] loadPrivateKey -> DEBU 033 Loading private key [dfb17cf51dc061d585b4850599be0e4b8b7cc8cc363a67c23bc03c6c5393b0e0] at [/etc/Hyperledger/peer/msp/keystore/dfb17cf51dc061d585b4850599be0e4b8b7cc8cc363a67c23bc03c6c5393b0e0_sk]...
2019-04-19 08:45:34.868 UTC [msp/identity] newIdentity -> DEBU 034 Creating identity instance for cert -----BEGIN CERTIFICATE-----
MIICGTCCAb+gAwIBAgIQTx2TvwYtAf62KKQliP6UoTAKBggqhkjOPQQDAjBzMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1UEBxMNU2FuIEZy
YW5jaXNjbzEZMBcGA1UEChMQb3JnMS5leGFtcGxlLmNvbTEcMBoGA1UEAxMTY2Eu
b3JnMS5leGFtcGxlLmNvbTAeFw0xNzA2MjYxMjQ5MjZaFw0yNzA2MjQxMjQ5MjZa
MFsxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1T
YW4gRnJhbmNpc2NvMR8wHQYDVQQDExZwZWVyMC5vcmcxLmV4YW1wbGUuY29tMFkw
EwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEPPHUp7+EYb2xIbleWfRClMgqbtQqRmIS
2a5F8T0L3J6IZp9wm7K+w4LIBIgw1Cz9D8nqHW6f4OYBrbp0cSGnR6NNMEswDgYD
VR0PAQH/BAQDAgeAMAwGA1UdEwEB/wQCMAAwKwYDVR0jBCQwIoAgGatlq7sEgH2t
EuTAqaqmZJ5who46vQIXoyLYnkfhpq4wCgYIKoZIzj0EAwIDSAAwRQIhAK4i2Hz2
K398TvjJk62neDoenYhkMY7rBN3BN/GI0G0SAiAOTx36wuy9/4BBV8NVBCZ9V+Iw
msdI9CyZ59oVMVmNYQ==
-----END CERTIFICATE-----
2019-04-19 08:45:34.869 UTC [msp] setupSigningIdentity -> DEBU 035 Signing identity expires at 2027-06-24 12:49:26 +0000 UTC
2019-04-19 08:45:34.869 UTC [msp] Validate -> DEBU 036 MSP Org1MSP validating identity

2019-04-19 08:45:34.871 UTC [msp] GetDefaultSigningIdentity -> DEBU 037 Obtaining default signing identity

2019-04-19 08:45:34.871 UTC [grpc] Printf -> DEBU 038 parsed scheme: ""

2019-04-19 08:45:34.871 UTC [grpc] Printf -> DEBU 039 scheme "" not registered, fallback to default scheme

2019-04-19 08:45:34.871 UTC [grpc] Printf -> DEBU 03a ccResolverWrapper: sending new addresses to cc: [{orderer.example.com:7050 0 <nil>}]

2019-04-19 08:45:34.871 UTC [grpc] Printf -> DEBU 03b ClientConn switching balancer to "pick_first"

2019-04-19 08:45:34.872 UTC [grpc] Printf -> DEBU 03c pickfirstBalancer: HandleSubConnStateChange: 0xc4203b44f0, CONNECTING

2019-04-19 08:45:34.876 UTC [grpc] Printf -> DEBU 03d pickfirstBalancer: HandleSubConnStateChange: 0xc4203b44f0, READY

2019-04-19 08:45:34.877 UTC [channelCmd] InitCmdFactory -> INFO 03e Endorser and orderer connections initialized

2019-04-19 08:45:34.886 UTC [msp] GetDefaultSigningIdentity -> DEBU 03f Obtaining default signing identity

2019-04-19 08:45:34.886 UTC [msp] GetDefaultSigningIdentity -> DEBU 040 Obtaining default signing identity

2019-04-19 08:45:34.886 UTC [msp/identity] Sign -> DEBU 041 Sign: plaintext: 0AA2060A074F7267314D53501296062D...6D706F736572436F6E736F727469756D

2019-04-19 08:45:34.886 UTC [msp/identity] Sign -> DEBU 042 Sign: digest: BF61ADBDFADA4F8F1D89F8DF9F79311ED38DD7C2D0A588724E5C1B0B2260E9EF

2019-04-19 08:45:34.887 UTC [msp] GetDefaultSigningIdentity -> DEBU 043 Obtaining default signing identity

2019-04-19 08:45:34.887 UTC [msp] GetDefaultSigningIdentity -> DEBU 044 Obtaining default signing identity

2019-04-19 08:45:34.887 UTC [msp/identity] Sign -> DEBU 045 Sign: plaintext: 0ADF060A1B08021A0608AE95E6E50522...41DCBDA3B6B3CE0380457C5C31A70492

2019-04-19 08:45:34.887 UTC [msp/identity] Sign -> DEBU 046 Sign: digest: D98584BA57E4D1E84AB18C605F682971C826F03AF8061AB290A2402BF4F00B19

2019-04-19 08:45:34.887 UTC [grpc] Printf -> DEBU 047 parsed scheme: ""

2019-04-19 08:45:34.888 UTC [grpc] Printf -> DEBU 048 scheme "" not registered, fallback to default scheme

2019-04-19 08:45:34.888 UTC [grpc] Printf -> DEBU 049 ccResolverWrapper: sending new addresses to cc: [{orderer.example.com:7050 0 <nil>}]

2019-04-19 08:45:34.888 UTC [grpc] Printf -> DEBU 04a ClientConn switching balancer to "pick_first"

2019-04-19 08:45:34.888 UTC [grpc] Printf -> DEBU 04b pickfirstBalancer: HandleSubConnStateChange: 0xc4203b4cf0, CONNECTING

2019-04-19 08:45:34.894 UTC [grpc] Printf -> DEBU 04c pickfirstBalancer: HandleSubConnStateChange: 0xc4203b4cf0, READY

2019-04-19 08:45:35.045 UTC [msp] GetDefaultSigningIdentity -> DEBU 04d Obtaining default signing identity

2019-04-19 08:45:35.045 UTC [msp] GetDefaultSigningIdentity -> DEBU 04e Obtaining default signing identity

2019-04-19 08:45:35.046 UTC [msp/identity] Sign -> DEBU 04f Sign: plaintext: 0ADF060A1B08051A0608AF95E6E50522...ECB292DBD39812080A021A0012021A00

2019-04-19 08:45:35.046 UTC [msp/identity] Sign -> DEBU 050 Sign: digest: 71152EAED821D462BC6FE14DBFA2F50C594C86736161B032C61D33A32A17B59F

2019-04-19 08:45:35.047 UTC [cli/common] readBlock -> INFO 051 Got status: &{NOT_FOUND}

2019-04-19 08:45:35.048 UTC [msp] GetDefaultSigningIdentity -> DEBU 052 Obtaining default signing identity

2019-04-19 08:45:35.049 UTC [grpc] Printf -> DEBU 053 parsed scheme: ""

2019-04-19 08:45:35.049 UTC [grpc] Printf -> DEBU 054 scheme "" not registered, fallback to default scheme

2019-04-19 08:45:35.049 UTC [grpc] Printf -> DEBU 055 ccResolverWrapper: sending new addresses to cc: [{orderer.example.com:7050 0 <nil>}]

2019-04-19 08:45:35.049 UTC [grpc] Printf -> DEBU 056 ClientConn switching balancer to "pick_first"

2019-04-19 08:45:35.050 UTC [grpc] Printf -> DEBU 057 pickfirstBalancer: HandleSubConnStateChange: 0xc4203ca3e0, CONNECTING

2019-04-19 08:45:35.057 UTC [grpc] Printf -> DEBU 058 pickfirstBalancer: HandleSubConnStateChange: 0xc4203ca3e0, READY

2019-04-19 08:45:35.059 UTC [channelCmd] InitCmdFactory -> INFO 059 Endorser and orderer connections initialized

2019-04-19 08:45:35.261 UTC [msp] GetDefaultSigningIdentity -> DEBU 05a Obtaining default signing identity

2019-04-19 08:45:35.261 UTC [msp] GetDefaultSigningIdentity -> DEBU 05b Obtaining default signing identity

2019-04-19 08:45:35.261 UTC [msp/identity] Sign -> DEBU 05c Sign: plaintext: 0ADF060A1B08051A0608AF95E6E50522...882803924437120 80A021A0012021A00

2019-04-19 08:45:35.261 UTC [msp/identity] Sign -> DEBU 05d Sign: digest: 78C8231BD1394ADEDE7E0B277AB2B11D512890331AC163A91F1FF28A091EF034

2019-04-19 08:45:35.275 UTC [cli/common] readBlock -> INFO 05e Received block: 0

2019-04-19 08:45:35.924 UTC [viperutil] getKeysRecursively -> DEBU 001 Found map[string]interface{} value for peer.BCCSP

2019-04-19 08:45:35.925 UTC [viperutil] unmarshalJSON -> DEBU 002 Unmarshal JSON: value cannot be unmarshalled: invalid character 'S' looking for beginning of value

2019-04-19 08:45:35.925 UTC [viperutil] getKeysRecursively -> DEBU 003 Found real value for peer.BCCSP.Default setting to string SW

2019-04-19 08:45:35.926 UTC [viperutil] getKeysRecursively -> DEBU 004 Found map[string]interface{} value for peer.BCCSP.SW

2019-04-19 08:45:35.927 UTC [viperutil] getKeysRecursively -> DEBU 005 Found map[string]interface{} value for peer.BCCSP.SW.FileKeyStore

2019-04-19 08:45:35.927 UTC [viperutil] unmarshalJSON -> DEBU 006 Unmarshal JSON: value cannot be unmarshalled: unexpected end of JSON input

2019-04-19 08:45:35.927 UTC [viperutil] getKeysRecursively -> DEBU 007 Found real value for peer.BCCSP.SW.FileKeyStore.KeyStore setting to string

2019-04-19 08:45:35.929 UTC [viperutil] unmarshalJSON -> DEBU 008 Unmarshal JSON: value cannot be unmarshalled: invalid character 'S' looking for beginning of value

2019-04-19 08:45:35.929 UTC [viperutil] getKeysRecursively -> DEBU 009 Found real value for peer.BCCSP.SW.Hash setting to string SHA2

2019-04-19 08:45:35.930 UTC [viperutil] unmarshalJSON -> DEBU 00a Unmarshal JSON: value is not a string: 256

2019-04-19 08:45:35.930 UTC [viperutil] getKeysRecursively -> DEBU 00b Found real value for peer.BCCSP.SW.Security setting to int 256

2019-04-19 08:45:35.931 UTC [viperutil] getKeysRecursively -> DEBU 00c Found map[string]interface{} value for peer.BCCSP.PKCS11

2019-04-19 08:45:35.932 UTC [viperutil] unmarshalJSON -> DEBU 00d Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:35.933 UTC [viperutil] getKeysRecursively -> DEBU 00e Found real value for peer.BCCSP.PKCS11.Hash setting to <nil> <nil>

2019-04-19 08:45:35.933 UTC [viperutil] unmarshalJSON -> DEBU 00f Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:35.934 UTC [viperutil] getKeysRecursively -> DEBU 010 Found real value for peer.BCCSP.PKCS11.Security setting to <nil> <nil>

2019-04-19 08:45:35.935 UTC [viperutil] getKeysRecursively -> DEBU 011 Found map[string]interface{} value for peer.BCCSP.PKCS11.FileKeyStore

2019-04-19 08:45:35.936 UTC [viperutil] unmarshalJSON -> DEBU 012 Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:35.939 UTC [viperutil] getKeysRecursively -> DEBU 013 Found real value for peer.BCCSP.PKCS11.FileKeyStore.KeyStore setting to <nil> <nil>

2019-04-19 08:45:35.940 UTC [viperutil] unmarshalJSON -> DEBU 014 Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:35.941 UTC [viperutil] getKeysRecursively -> DEBU 015 Found real value for peer.BCCSP.PKCS11.Library setting to <nil> <nil>

2019-04-19 08:45:35.942 UTC [viperutil] unmarshalJSON -> DEBU 016 Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:35.943 UTC [viperutil] getKeysRecursively -> DEBU 017 Found real value for peer.BCCSP.PKCS11.Label setting to <nil> <nil>

2019-04-19 08:45:35.943 UTC [viperutil] unmarshalJSON -> DEBU 018 Unmarshal JSON: value is not a string: <nil>

2019-04-19 08:45:35.944 UTC [viperutil] getKeysRecursively -> DEBU 019 Found real value for peer.BCCSP.PKCS11.Pin setting to <nil> <nil>

2019-04-19 08:45:35.944 UTC [viperutil] EnhancedExactUnmarshalKey -> DEBU 01a map[peer.BCCSP:map[Default:SW SW:map[FileKeyStore:map[KeyStore:] Hash:SHA2 Security:256] PKCS11:map[Label:<nil> Pin:<nil> Hash:<nil> Security:<nil> FileKeyStore:map[KeyStore:<nil>] Library:<nil>]]]

2019-04-19 08:45:35.950 UTC [bccsp_sw] openKeyStore -> DEBU 01b KeyStore opened at [/etc/Hyperledger/msp/users/Admin@org1.example.com/msp/keystore]...done

2019-04-19 08:45:35.951 UTC [bccsp] initBCCSP -> DEBU 01c Initialize BCCSP [SW]

2019-04-19 08:45:35.952 UTC [msp] getPemMaterialFromDir -> DEBU 01d Reading directory /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/signcerts

2019-04-19 08:45:35.958 UTC [msp] getPemMaterialFromDir -> DEBU 01e Inspecting file /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/signcerts/Admin@org1.example.com-cert.pem

2019-04-19 08:45:35.962 UTC [msp] getPemMaterialFromDir -> DEBU 01f Reading directory /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/cacerts

2019-04-19 08:45:35.964 UTC [msp] getPemMaterialFromDir -> DEBU 020 Inspecting file /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/cacerts/ca.org1.example.com-cert.pem

2019-04-19 08:45:35.967 UTC [msp] getPemMaterialFromDir -> DEBU 021 Reading directory /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/admincerts

2019-04-19 08:45:35.974 UTC [msp] getPemMaterialFromDir -> DEBU 022 Inspecting file /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/admincerts/Admin@org1.example.com-cert.pem

2019-04-19 08:45:35.979 UTC [msp] getPemMaterialFromDir -> DEBU 023 Reading directory /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/intermediatecerts

2019-04-19 08:45:35.979 UTC [msp] getMspConfig -> DEBU 024 Intermediate certs folder not found at [/etc/Hyperledger/msp/users/Admin@org1.example.com/msp/intermediatecerts]. Skipping. [stat /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/intermediatecerts: no such file or directory]

2019-04-19 08:45:35.979 UTC [msp] getPemMaterialFromDir -> DEBU 025 Reading directory /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/tlscacerts

2019-04-19 08:45:35.985 UTC [msp] getPemMaterialFromDir -> DEBU 026 Inspecting file /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/tlscacerts/tlsca.org1.example.com-cert.pem

2019-04-19 08:45:35.990 UTC [msp] getPemMaterialFromDir -> DEBU 027 Reading directory /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/tlsintermediatecerts

2019-04-19 08:45:35.990 UTC [msp] getMspConfig -> DEBU 028 TLS intermediate certs folder not found at [/etc/Hyperledger/msp/users/Admin@org1.example.com/msp/tlsintermediatecerts]. Skipping. [stat /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/tlsintermediatecerts: no such file or directory]

2019-04-19 08:45:35.990 UTC [msp] getPemMaterialFromDir -> DEBU 029 Reading directory /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/crls

2019-04-19 08:45:35.990 UTC [msp] getMspConfig -> DEBU 02a crls folder not found at [/etc/Hyperledger/msp/users/Admin@org1.example.com/msp/crls]. Skipping. [stat /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/crls: no such file or directory]

2019-04-19 08:45:35.990 UTC [msp] getMspConfig -> DEBU 02b MSP configuration file not found at [/etc/Hyperledger/msp/users/Admin@org1.example.com/msp/config.yaml]: [stat /etc/Hyperledger/msp/users/Admin@org1.example.com/msp/config.yaml: no such file or directory]

2019-04-19 08:45:35.991 UTC [msp] newBccspMsp -> DEBU 02c Creating BCCSP-based MSP instance

2019-04-19 08:45:35.991 UTC [msp] New -> DEBU 02d Creating Cache-MSP instance

2019-04-19 08:45:35.991 UTC [msp] loadLocaMSP -> DEBU 02e Created new local MSP

2019-04-19 08:45:35.991 UTC [msp] Setup -> DEBU 02f Setting up MSP instance Org1MSP

2019-04-19 08:45:35.992 UTC [msp/identity] newIdentity -> DEBU 030 Creating identity instance for cert -----BEGIN CERTIFICATE-----

MIICQjCCAemgAwIBAgIQDJbOh88U+tlJ9He5sjUwBDAKBggqhkjOPQQDAjBzMQsw
CQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1UEBxMNU2FuIEZy
YW5jaXNjbzEZMBcGA1UEChMQb3JnMS5leGFtcGxlLmNvbTEcMBoGA1UEAxMTY2Eu
b3JnMS5leGFtcGxlLmNvbTAeFw0xNzA2MjYxMjQ5MjZaFw0yNzA2MjQxMjQ5MjZa
MHMxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlhMRYwFAYDVQQHEw1T
YW4gRnJhbmNpc2NvMRkwFwYDVQQKExBvcmcxLmV4YW1wbGUuY29tMRwwGgYDVQQD
ExNjYS5vcmcxLmV4YW1wbGUuY29tMFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE
kQ216YBD8kks+IzKJyBmdLqh/L/sEZ5twTqYpsK1ztNhIUDplsletAFOJQWRH+kb
hzFFNvS1qwveGRU6ztN5haNfMF0wDgYDVR0PAQH/BAQDAgGmMA8GA1UdJQQIMAYG
BFUdJQAwDwYDVR0TAQH/BAUwAwEB/zApBgNVHQ4EIgQgGatlq7sEgH2tEuTAqaqm
ZJ5who46vQIXoyLYnkfhpq4wCgYIKoZIzj0EAwIDRwAwRAIgCyrj/lUjtBYaEgMt
x98l5z+iLU6r+gp4CsdcDYzKLugCIGXlcU56avWSUtRAGn8Avpb6TOxtkrKIpeTE
QfM8VsS/
-----END CERTIFICATE-----

2019-04-19 08:45:35.992 UTC [msp/identity] newIdentity -> DEBU 031 Creating identity instance for cert -----BEGIN CERTIFICATE-----

MIICGjCCAcCgAwIBAgIRANuOnVN+yd/BGyoX7ioEklQwCgYIKoZIzj0EAwIwczEL
MAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNhbiBG
cmFuY2lzY28xGTAXBgNVBAoTEG9yZzEuZXhhbXBsZS5jb20xHDAaBgNVBAMTE2Nh
Lm9yZzEuZXhhbXBsZS5jb20wHhcNMTcwNjI2MTI0OTI2WhcNMjcwNjI0MTI0OTI2
WjBbMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1UEBxMN

U2FuIEZyYW5jaXNjbzEfMB0GA1UEAwwWQWRtaW5Ab3JnMS5leGFtcGxlLmNvbTBZ
MBMGByqGSM49AgEGCCqGSM49AwEHA0IABGu8KxBQ1GkxSTMVoLv7NXiYKWj5t6
Dh
WRTJBHnLkWV7lRUfYaKAKFadSii5M7Z7ZpwD8NS7IsMdPR6Z4EyGgwKjTTBLMA4G
A1UdDwEB/wQEAwIHgDAMBgNVHRMBAf8EAjAAMCsGA1UdIwQkMCKAIBmrZau7BI
B9
rRLkwKmqpmSecIaOOr0CF6Mi2J5H4aauMAoGCCqGSM49BAMCA0gAMEUCIQC4sKQ6
CEgqbTYe48az95W9/hnZ+7DI5eSnWUwV9vCd/gIgS5K6omNJydoFoEpaEIwM97uS
XVMHPa0iyC497vdNURA=
-----END CERTIFICATE-----
2019-04-19 08:45:36.049 UTC [msp/identity] newIdentity -> DEBU 032 Creating identity instance for cert -----BEGIN CERTIFICATE-----
MIICGjCCAcCgAwIBAgIRANuOnVN+yd/BGyoX7ioEklQwCgYIKoZIzj0EAwIwczEL
MAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNhbiB
G
cmFuY2lzY28xGTAXBgNVBAoTEG9yZzEuZXhhbXBsZS5jb20xHDAaBgNVBAMTE2Nh
Lm9yZzEuZXhhbXBsZS5jb20wHhcNMTcwNjI2MTI0OTI2WhcNMjcwNjI0MTI0OTI2
WjBbMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1U
EBxMN
U2FuIEZyYW5jaXNjbzEfMB0GA1UEAwwWQWRtaW5Ab3JnMS5leGFtcGxlLmNvbTBZ
MBMGByqGSM49AgEGCCqGSM49AwEHA0IABGu8KxBQ1GkxSTMVoLv7NXiYKWj5t6
Dh
WRTJBHnLkWV7lRUfYaKAKFadSii5M7Z7ZpwD8NS7IsMdPR6Z4EyGgwKjTTBLMA4G
A1UdDwEB/wQEAwIHgDAMBgNVHRMBAf8EAjAAMCsGA1UdIwQkMCKAIBmrZau7BI
B9
rRLkwKmqpmSecIaOOr0CF6Mi2J5H4aauMAoGCCqGSM49BAMCA0gAMEUCIQC4sKQ6
CEgqbTYe48az95W9/hnZ+7DI5eSnWUwV9vCd/gIgS5K6omNJydoFoEpaEIwM97uS
XVMHPa0iyC497vdNURA=
-----END CERTIFICATE-----
2019-04-19 08:45:36.050 UTC [bccsp_sw] loadPrivateKey -> DEBU 033 Loading private key [114aab0e76bf0c78308f89efc4b8c9423e31568da0c340ca187a9b17aa9a4457] at

[/etc/Hyperledger/msp/users/Admin@org1.example.com/msp/keystore/114aab0e76bf0c78308f89efc4b8c9423e31568da0c340ca187a9b17aa9a4457_sk]...

2019-04-19 08:45:36.056 UTC [msp/identity] newIdentity -> DEBU 034 Creating identity instance for cert -----BEGIN CERTIFICATE-----

MIICGjCCAcCgAwIBAgIRANuOnVN+yd/BGyoX7ioEklQwCgYIKoZIzj0EAwIwczEL
MAkGA1UEBhMCVVMxEzARBgNVBAgTCkNhbGlmb3JuaWExFjAUBgNVBAcTDVNhbiBG
cmFuY2lzY28xGTAXBgNVBAoTEG9yZzEuZXhhbXBsZS5jb20xHDAaBgNVBAMTE2Nh
Lm9yZzEuZXhhbXBsZS5jb20wHhcNMTcwNjI2MTI0OTI2WhcNMjcwNjI0MTI0OTI2
WjBbMQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTEWMBQGA1UEBxMN
U2FuIEZyYW5jaXNjbzEfMB0GA1UEAwwWQWRtaW5Ab3JnMS5leGFtcGxlLmNvbTBZ
MBMGByqGSM49AgEGCCqGSM49AwEHA0IABGu8KxBQ1GkxSTMVoLv7NXiYKWj5t6Dh
WRTJBHnLkWV7lRUfYaKAKFadSii5M7Z7ZpwD8NS7IsMdPR6Z4EyGgwKjTTBLMA4G
A1UdDwEB/wQEAwIHgDAMBgNVHRMBAf8EAjAAMCsGA1UdIwQkMCKAIBmrZau7BIB9
rRLkwKmqpmSecIaOOr0CF6Mi2J5H4aauMAoGCCqGSM49BAMCA0gAMEUCIQC4sKQ6
CEgqbTYe48az95W9/hnZ+7DI5eSnWUwV9vCd/gIgS5K6omNJydoFoEpaEIwM97uS
XVMHPa0iyC497vdNURA=

-----END CERTIFICATE-----

2019-04-19 08:45:36.057 UTC [msp] setupSigningIdentity -> DEBU 035 Signing identity expires at 2027-06-24 12:49:26 +0000 UTC

2019-04-19 08:45:36.058 UTC [msp] Validate -> DEBU 036 MSP Org1MSP validating identity

2019-04-19 08:45:36.061 UTC [msp] GetDefaultSigningIdentity -> DEBU 037 Obtaining default signing identity

2019-04-19 08:45:36.063 UTC [grpc] Printf -> DEBU 038 parsed scheme: ""

2019-04-19 08:45:36.063 UTC [grpc] Printf -> DEBU 039 scheme "" not registered, fallback to default scheme

2019-04-19 08:45:36.064 UTC [grpc] Printf -> DEBU 03a ccResolverWrapper: sending new addresses to cc: [{peer0.org1.example.com:7051 0 <nil>}]

2019-04-19 08:45:36.065 UTC [grpc] Printf -> DEBU 03b ClientConn switching balancer to "pick_first"

2019-04-19 08:45:36.065 UTC [grpc] Printf -> DEBU 03c pickfirstBalancer: HandleSubConnStateChange: 0xc42027a480, CONNECTING

2019-04-19 08:45:36.070 UTC [grpc] Printf -> DEBU 03d pickfirstBalancer: HandleSubConnStateChange: 0xc42027a480, READY

2019-04-19 08:45:36.070 UTC [channelCmd] InitCmdFactory -> INFO 03e Endorser and orderer connections initialized

2019-04-19 08:45:36.073 UTC [msp/identity] Sign -> DEBU 03f Sign: plaintext: 0A9F070A5B08011A0B08B095E6E50510...4293E25106B71A080A000A000A000A00

2019-04-19 08:45:36.073 UTC [msp/identity] Sign -> DEBU 040 Sign: digest: 22F34FF2EB12CD9CA3FA1C79B8D0E921DBE5C2AA3787EE9D5F21F903C52F2A75

2019-04-19 08:45:37.052 UTC [channelCmd] executeJoin -> INFO 041 Successfully submitted proposal to join channel

user@snf-1559:~/project/fabric-dev-servers$

APPENDIX B: Project Activity Schedule

Items of Work/Activities												
	1	2	3	4	5	6	7	8	9	10	11	12
Proposal concept writing and submission												
Writing of literature review and Methodology												
Proposal Defense												
Preparation for and data collection												
Data analysis and interpretation												
System Development and testing												
Thesis writing												
Submission of draft thesis for review												
Submission of final thesis to SGS												
Thesis defense												
Thesis corrections and final submission												

APPENDIX C: Turnit in Similarity Index