

**DEVELOPING A BLOCKCHAIN APPLICATION TO STORE PUBLIC RECORDS.**

BY

NJOROGE NAHASHON NJENGA

IF/20/14

**PROJECT REPORT SUBMITTED TO THE SCHOOL OF INFORMATION SCIENCES IN PARTIAL FULFILLMENT OF THE REQUIREMENTS OF THE DEGREE OF BACHELOR OF SCIENCE IN INFORMATION SCIENCE OF MOI UNIVERSITY**

AUGUST 2018

# DECLARATION

I declare that this is my original work and has not been presented in any other school for examination purposes.

Name: NAHASHON NJENGA

Signature Date

Supervisor:

Name: MR. NICHOLAS KIGET

Signature Date

# ACKNOWLEDGEMENT

Enough cannot be said to thank our peer reviewers who helped mold the final form of this project, from catching simple bugs to helping with the mental design of the implementation details.

Special thanks to Mr. Dan Samoei, who helped supervise this project and make sure it meets all academic requirements.

I also want to thank my brother John Muiruri and Damaris Simiyu, for providing the necessary documents like log book, title deed and driving license. These documents were relevant in making the system store all the required information.

I cannot forget to thank my classmates and roommate for believing in me. That encouraged me to push forward with this mammoth of an undertaking.

# ABSTRACT

Fraud in the public sector has contributed to the deep rooted corruption we see all the time in the news. Ranging from land grabbing, to use of fake academic certificates and theft, illegal sale and purchasing of vehicles. People seem to have found illegal means to generate certificates and other documents that seem authentic, and use them in the economy.

The government has gone a step further to digitize the process of acquiring these documents. This has helped accelerate the process of acquiring these documents significantly. But the issue of fraud still looms. This is because of the nature of the technologies used. Normal databases offer a way for someone to edit any records without a trace. This ideally means a person with a contact to the system administrator can easily bribe them or worse still blackmail them to edit their records.

The blockchain helps solve this problem. It promises immutability; once records are added to the blockchain, there is no changing them. And the ones that need to show change of ownership, like logbooks and title deeds, a trail of various owners over time can be obtained.

**TABLE OF CONTENTS**

Contents

[DECLARATION i](#_Toc522397899)

[ACKNOWLEDGEMENT ii](#_Toc522397900)

[ABSTRACT iii](#_Toc522397901)

[+LIST OF FIGURES AND TABLES vi](#_Toc522397902)

[LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS vii](#_Toc522397903)

[CHAPTER ONE 1](#_Toc522397904)

[INTRODUCTION 1](#_Toc522397905)

[1.1 Chapter Overview 1](#_Toc522397906)

[1.2 Background 1](#_Toc522397907)

[1.3 Statement of the problem 2](#_Toc522397908)

[1.4 Aim and Objectives of the study 3](#_Toc522397909)

[1.4.1 Aim of the study 3](#_Toc522397910)

[1.4.2 Objective of the study 3](#_Toc522397911)

[1.5 Research Questions 3](#_Toc522397912)

[CHAPTER TWO 6](#_Toc522397913)

[LITERATURE REVIEW 6](#_Toc522397914)

[2.3.1 Origin 14](#_Toc522397915)

[2.3.2 Takeover by Google 14](#_Toc522397916)

[2.3.3 Different Versions of Android OS 15](#_Toc522397917)

[Android Pros 17](#_Toc522397918)

[Android Cons 17](#_Toc522397919)

[2.4.1 United States of America 19](#_Toc522397920)

[2.4.2 Europe 20](#_Toc522397921)

[2.4.3 Current state of things globally when it comes to shopping 21](#_Toc522397922)

[2.5 Why Technology is Being Integrated in the Shopping Sector 21](#_Toc522397923)

[2.5.1 Benefits of integrating mobile applications in e-commerce 22](#_Toc522397924)

[Keeping up with competition 23](#_Toc522397925)

[2.6 A Look At Past And Present Shopping Systems Already In Use 25](#_Toc522397926)

[2.6.1 Globally 25](#_Toc522397927)

[2.6.2 Current shopping Systems in Kenya 33](#_Toc522397928)

[2.6.3 How developed system will complement existing Shopping systems 34](#_Toc522397929)

[2.7 Chapter Summary 35](#_Toc522397930)

[CHAPTER 3 35](#_Toc522397931)

[METHODOLOGY 35](#_Toc522397932)

[3.1 Chapter Overview 35](#_Toc522397933)

[3.2.1 The Waterfall Model 36](#_Toc522397934)

[3.2.2 Rapid Application Development 37](#_Toc522397935)

[3.3 Data Collection Instruments 39](#_Toc522397936)

[3.3.1 Questionnaire 40](#_Toc522397937)

[3.3.2 Interview 40](#_Toc522397938)

[3.3.3 Observation 40](#_Toc522397939)

[3.5 Study Area 40](#_Toc522397940)

[3.6 Target Population 41](#_Toc522397941)

[3.7 Sampling size and sampling procedures 41](#_Toc522397942)

[3.8 Data Collection Procedures 42](#_Toc522397943)

[CHAPTER 4 43](#_Toc522397944)

[SYSTEM ANALYSIS AND DESIGN 43](#_Toc522397945)

[4.2.1 Overview of the Proposed System 43](#_Toc522397946)

[4.2.2 Intended users 46](#_Toc522397947)

[4.2.3 Proposed system functions 46](#_Toc522397948)

[4.2.4 Functional requirements 47](#_Toc522397949)

[4.2.5 Non-Functional requirement 49](#_Toc522397950)

[4.3 System Design 50](#_Toc522397951)

[4.3.1 Structure Design 50](#_Toc522397952)

[4.4 Smart Shopping Mobile Application System Implementation 59](#_Toc522397953)

[4.4.1 Tools used 59](#_Toc522397954)

[4.5 Testing 67](#_Toc522397955)

[4.5.1 Testing Strategies 68](#_Toc522397956)

[4.6 Chapter Summary 69](#_Toc522397957)

[CHAPTER FIVE 70](#_Toc522397958)

[SUMMARY, CONCLUSION AND RECOMMENDATIONS 70](#_Toc522397959)

[5.1 Chapter Overview 70](#_Toc522397960)

[5.2 Summary of Findings 70](#_Toc522397961)

[5.3 Conclusion 71](#_Toc522397962)

[5.4 Recommendation 71](#_Toc522397963)

[5.5 Suggestion for further research 72](#_Toc522397964)

[REFERENCES 72](#_Toc522397965)

[APPENDICES 75](#_Toc522397966)

[Appendix 1: Letter to respondent/ site manager 75](#_Toc522397967)

[Appendix 2: Questionnaire 76](#_Toc522397968)

[Appendix 3: System requirements 79](#_Toc522397969)

[Appendix 4: Project Codes 79](#_Toc522397970)

# LIST OF FIGURES AND TABLES

[Figure 2.1 7](#_Toc522387328)

[Figure 2.2 8](#_Toc522387329)

[Figure 2.3 9](#_Toc522387330)

[Figure 2.4 9](#_Toc522387331)

[Figure 2.5 10](#_Toc522387332)

[Figure 2.6 10](#_Toc522387333)

[Figure 2.7 11](#_Toc522387334)

[Figure 2.8 12](#_Toc522387335)

[Figure 2.9 12](#_Toc522387336)

[Figure 2.10 22](#_Toc522387337)

[Figure 2.11 26](#_Toc522387338)

[Figure 2.12 26](#_Toc522387339)

[Figure 2.13 27](#_Toc522387340)

[Figure 2.14 28](#_Toc522387341)

[Figure 2.15 29](#_Toc522387342)

[Figure 2.16 29](#_Toc522387343)

[Figure 2.17 30](#_Toc522387344)

[Figure 2.18 31](#_Toc522387345)

[Figure 2.19 31](#_Toc522387346)

[Figure 2.20 32](#_Toc522387347)

[Figure 3.1 Waterfall model 36](#_Toc522387348)

[**Figure 3.2**: Prototyping approach used by RAD. 37](#_Toc522387349)

[Table 3.1: Sampling Frame 41](#_Toc522387350)

[Table 3.2: Sampling Design 41](#_Toc522387351)

[Figure 4.1 System’s architectural design 44](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387352)

[Figure 4.2 Data Flow Diagram 46](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387353)

[Figure 4.3 Use Case Diagram 52](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387354)

[Figure 4.4 Activity diagram for Administrator 53](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387355)

[Figure 4.5 Activity diagram for user 54](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387356)

[Figure 4.6 Sequence Diagram for User 57](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387357)

[Figure 4.7 Sequence Diagram for Administrator 58](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387358)

[Figure 4.8 Class Diagram 59](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387359)

[Figure 4.9 Home tab 61](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387360)

[Figure 4.10 Amount tab 62](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387361)

[Figure 4.11 Barcode Scanner tab 63](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387362)

[Figure 4.12 Scanned List tab 64](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387363)

[Figure 4.13 Mobile payment tab 65](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387364)

[Figure 4.14 Shopping List tab 66](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387365)

[Figure 4.15 Calculator Tab 67](file:///C:\Users\freda\Desktop\Project%20Documentation\FINAL%20PROJ%20FOLDER2\Project%20DocumentationZXA.docx#_Toc522387366)

# LIST OF TERMS AND ABBREVIATIONS

**API -** Application Programming Interface

**PoW -**Proof of Work.

**Blockchain** - a peer-to-peer distributed ledger that is cryptographically secure, append-only, immutable and updateable only via consensus.

**Transaction** - Any action which modifies the state of your contract storage (and therefore, the blockchain in general) will occur as a transaction. Essentially a Create, Update or Delete operation in normal databases.

**Smart Contract**- this is a secure and unstoppable computer program representing an agreement that is automatically executable and enforceable.

**Wallet** - A **blockchain wallet** is a digital **wallet** that allows users to manage cryptocurrency.

**Mining** - **mining** is the process of adding transaction records to a blockchain.

# CHAPTER ONE

# INTRODUCTION

## 1.1 Chapter Overview

This chapter outlines and presents the background of the study, the problem statement, aim and objectives of the study, assumptions, significance and scope of the study.

## 1.2 Background

In 2017, my brother was running a car hire business. The business had been doing great, until the worst happened. A client rent the car for a week, and in the middle of the week, the client calls him notifying him that his car had been stolen. It turns out that this client went and lent out the car to somebody else who turns out to be a thief. The thief went and processed a fake log book and was in the process of selling the car to another person, when my brother received the news and turned off the car. With rental cars, and even most personal cars, you ought to install a car tracker in it, which helps you track the location of the car in real time on google maps, and also enables you to turn off the car remotely. My brother then went to the police to report this incident. And after a long process, he finally got his car back. The car tracker played a significant role in helping recover this lost car. If the car did not have this tracker, it would have taken a very long time to recover this car. This goes to show how easy it is to generate fake documents to be used for transactions in the country. If the system am proposing was in operation, the buyer would have simply gone to the site, searched the number plate of the car, and gotten the authentic owner of the car, and would have realized the seller is not the owner. This is just one case of many that go undocumented in our country. We need to come up with measures to help curb corruption and fraud in Kenya.

## 1.3 Statement of the problem

Over the years, corruption has continued to plague the Kenyan economy. Every newly elected government promises to come up with a solution. From the national government to county governments, measures such as lifestyle audits have been conducted to try and smoke out the corrupt.

We cannot have a ‘one size fits all’ mentality when it comes to solving the corruption problem. We have to identify various ways in which corruption takes place. The following are the most common methods of corruption:

**Bribery**

Bribery involves the improper use of gifts and favours in exchange for personal gain. It is the common form of corruption. The types of favours given are diverse and may include money, gifts, sexual favours, company shares, entertainment, employment and political benefits. The personal gain that is given can be anything from actively giving preferential treatment to having an indiscretion or crime overlooked.

**Embezzlement, theft and fraud**

Embezzlement and theft involve someone with access to funds or assets illegally taking control of them. Fraud involves using deception to convince the owner of funds or assets to give them up to an unauthorized party.

Examples include the misdirection of company funds into "shadow companies" (and then into the pockets of corrupt employees), the skimming of foreign aid money, scams and other corrupt activity.

**Graft**

The political act of **Graft** is a well-known and now global form of political corruption, being the unscrupulous and illegal use of a politician's authority for personal gain, when funds intended for public projects are intentionally misdirected in order to maximize the benefits to illegally private interests of the corrupted individual(s) and their cronies.

**Extortion and blackmail**

While bribery is the use of positive inducements for corrupt aims, extortion and blackmail centre around the use of threats. This can be the threat of violence or false imprisonment as well as exposure of an individual's secrets or prior crimes.

This includes such behavior as an influential person threatening to go to the media if they do not receive speedy medical treatment (at the expense of other patients), threatening a public official with exposure of their secrets if they do not vote in a particular manner, or demanding money in exchange for continued secrecy.

**Influence peddling**

Influence peddling is the illegal practice of using one's influence in government or connections with persons in authority to obtain favours or preferential treatment for another, usually in return for payment.

**Networking**

Networking can be an effective way for job-seekers to gain a competitive edge over others in the job-market. The idea is to cultivate personal relationships with prospective employers, selection panelists, and others, in the hope that these personal affections will influence future hiring decisions. This form of networking has been described as an attempt to corrupt formal hiring processes, where all candidates are given an equal opportunity to demonstrate their merits to selectors. The networker is accused of seeking non-meritocratic advantage over other candidates; advantage that is based on personal fondness rather than on any objective appraisal of which candidate is most qualified for the position.

**Abuse of discretion**

Abuse of discretion refers to the misuse of one's powers and decision-making facilities. Examples include a judge improperly dismissing a criminal case or a customs official using their discretion to allow a banned substance through a port.

**Favouritism, nepotism and clientelism**

Favouritism, nepotism and clientelism involve the favouring of not the perpetrator of corruption but someone related to them, such as a friend, family member or member of an association. Examples would include hiring or promoting a family member or staff member to a role they are not qualified for, who belongs to the same political party as you, regardless of merit.

As mentioned earlier, we cannot have one solution to solve all the forms of corruption outlined above. But as a start we should automate as many government services as possible, and even go as far as to use technologies such as blockchain that promise immutability of records once created.

## 1.4 Aim and Objectives of the study

### 1.4.1 Aim of the study

The main aim of this project is to design and develop a blockchain application to store public records. The application is intended for use by the various government bodies like E-Citizen, Kenya Revenue Authority and Kenya National Examination Council to add records to the blockchain, and the general public to query that data.

### 1.4.2 Objective of the study

1. To adopt a suitable methodology and use it to develop a blockchain application.
2. To adopt a suitable blockchain technology stack and use it to develop this system
3. To perform an analysis on requirements gathered for the application system.
4. To provide a way of storage and retrieval of public records secure, convenient and effortless.

## 1.5 Research Questions

1. What methodologies will be used to model and develop the application system?
2. What are the user requirements for the application system?
3. How is the system making of storage and retrieval of public records secure, convenient and effortless?

1.6 Assumptions of the study

The research was based on the assumption that the government will integrate this system with existing systems through APIs (Application Programming Interface). This therefore means that this system is a back end system, with no user interface.

1.7 Significance of the study

There has been numerus proposals [1] to use the blockchain to solve various problems in many sectors of our society ranging from the financial sector, medical sector and insurance, to education sector and the government as a whole. The existing technologies have helped automate processes in these sectors, but have been limited by the underlying architecture.

This application system on completion will be important to country because it will integrate with the existing technologies seamlessly yet provide the benefits of the blockchain. This means the users at the affected sectors will need very little training to get up to speed with this system. The user interface there are using now to populate data to their database will not be affected, only the processing and storage will, which as it stands they do not know or care since it is mean for the technical people to understand. Migration of data will also be made seamless through the use of APIs. The public will also visit the normal websites they do to get these services, and they will not notice any change in the way things are working at the moment. What they will get as an addition is the sense of security that the records they are accessing are authentic and tamper proof.

1.8 Scope of the study

The blockchain is a form of database by itself. As a result the scope of the system will cover the storage and retrieval of records. How they are populated and displayed on the user’s side has already been taken care of by the existing systems, as discussed in chapter 2 of this documentation. User authentication and authorization is also outside the scope of this project. The implementation of this system assumes that data coming to the system is from authorized user accounts. In addition, it is assumed that the data has passed some form of validation on the client side. Additional validation is also provided in this system to ensure only valid data is stored.

1.9 Chapter Summary

This chapter presents the background of the study, the problem statement, main aim of the study, its objectives and research questions, assumptions of the study, significance and scope. In the next chapter, a theoretical framework and related literature will be reviewed.

# CHAPTER TWO

# LITERATURE REVIEW

2.1 Chapter Overview

This chapter focuses on the study and scrutiny of existing systems this system will integrate with. The main purpose for this is to discover what has been done before, what is being done presently and what will be done in the future in regard to storage and retrieval of public records, and how this system will fill some of the gaps left by the existing systems and complement them. The chapter will also look at the history of blockchain technology, the architecture behind it, its application, advantages, disadvantages and limitations of this technology. I shall also explain how this system will complement these current systems.

2.2 The History of Blockchain Technology

You can’t discuss the history of blockchain technology without first starting with a discussion about Bitcoin. When Satoshi Nakamoto, whose true identity is still unknown, released the whitepaper Bitcoin: A Peer to Peer Electronic Cash System [2] in 2008 that described a “purely peer-to-peer version of electronic cash” known as Bitcoin, blockchain technology made its public debut. Blockchain, the technology that runs Bitcoin, has developed over the last decade into one of today’s biggest ground-breaking technologies with potential to impact every industry from financial to manufacturing to educational institutions. Shortly after Nakamoto’s whitepaper was released, Bitcoin was offered up to the open source community in 2009. Blockchain provided the answer to digital trust because it records important information in a public space and doesn’t allow anyone to remove it. It’s transparent, time-stamped and decentralized.

Blockchain is to Bitcoin, what the internet is to email. A big electronic system, on top of which you can build applications. Currency is just one.

Sally Davies, Financial Times Technology reporter.

**Blockchain Separates from Bitcoin**

Even today, there are many who believe Bitcoin and blockchain are one and the same, even though they are not. Those who started to realize around 2014 that blockchain could be used for more than cryptocurrency started to invest in and explore how blockchain could alter many different kinds of operations. At its core, blockchain is an open, decentralized ledger that records transactions between two parties in a permanent way without needing third-party authentication. This creates an extremely efficient process and one people predict will dramatically reduce the cost of transactions.

When entrepreneurs understood the power of blockchain, there was a surge of investment and discovery to see how blockchain could impact supply chains, healthcare, insurance, transportation, voting, contract management and more. Nearly 15% of financial institutions [3] are currently using blockchain technology.

**Ethereum Rises: Smart Contracts**

Vitalik Buterin [4], co-founder of Ethereum and Bitcoin magazine, was also an initial contributor to the Bitcoin codebase, but became frustrated around 2013 with its programming limitations and pushed for a malleable blockchain. Met with resistance from the Bitcoin community, Buterin set out to build the second public blockchain called Ethereum. The largest difference between the two is that Ethereum can record other assets such as loans or contracts, not just currency. Ethereum launched in 2015 and can be used to build “smart contracts”—those that can automatically process based on a set of criteria established in the Ethereum blockchain. This technology has attracted the attention of corporations such as Microsoft, BBVA and UBS [5] who are intrigued by the potential of the smart contract functionality to save time and money.

**Transition to Proof of Stake**

Currently, blockchain operates on the proof of work concept where an expensive computer calculation or “mining” is done in order to create a block (or a new set of trustless transactions). Currently, when you initiate a transaction, it is bundled into a block. Then miners verify the transactions are legitimate within that block by solving a proof-of-work problem—a very difficult mathematical problem that takes an extraordinary amount of computing power to solve. The first miner to solve the problem gets a reward and then the verified transaction is stored on the blockchain. Ethereum developers are interested in changing to a new consensus system called proof of stake.

Proof of stake has the same goal as proof of work (PoW)—to validate transactions and achieve consensus in the chain—and it uses an algorithm but with a different process. With proof of stake, the creator of a new block “is chosen in a deterministic way, depending on its wealth, also defined as a stake.” [6] Since in a proof of stake system, there is no block reward, but the miners, known as forgers, get the transaction fees. Proponents of this shift, including Ethereum co-founder Buterin, like proof of stake for the energy and cost savings realized to get to a distributed form of consensus.

**Blockchain Scaling on the Horizon**

Since currently, every computer in a blockchain network processes every transaction, it can be very slow. A blockchain scaling solution would determine how many computers are necessary to validate every transaction in a way that doesn’t compromise security.

Today, Bitcoin is just one of the several hundred applications that use blockchain technology. It’s been an impressive decade of transformation for blockchain technology and it will be intriguing to see where the next decade takes us.

The blockchain is based on the distributed systems architecture.

***Distributed systems*** are a computing paradigm whereby two or more nodes work with each other in a coordinated fashion in order to achieve a common outcome and its modeled in such a way that end users see a single logical platform.

A***node***is an individual player in a distributed system.

Distributed systems are so challenging to design that a theorem known as CAP theorem has been proved and states that a distributed system cannot have all much desired properties simultaneously. i.e. *Any distributed system cannot have Consistency, Availability, and Partition tolerance simultaneously.*

**Generic elements of a blockchain**

**Addresses, public and private keys**

Addresses are unique identifiers that are used in a transaction on the blockchain to denote senders and recipients. An address is a public key. An address is just like an account number in bank. It denotes your unique identity on the blockchain from which you can transfer out cryptocurrency and to which you can receive cryptocurrency. In blockchain a standard address has something like 34 signs and in Ethereum it starts with a 0x. It may look something like 0xee2eadd6d2b262194ac81c229ad1487ec0e3d7cc.

Every address on the bitcoin blockchain comes attached with a Private key and a Public key. These together form the pillars of security in the Blockchain network. Private and Public keys always work in a pair. The private key is a string that looks the way the address does and is unique to the owner of a particular Bitcoin address. The owner of this Bitcoin address uses his private key to digitally sign any transaction that he makes. The public key is a long alphanumeric string that is generated by the private key to an account and this can be publicly shared so that miners can verify digitally signed transactions. As the name suggests, a user’s private key is private to the user and the public key is known to everyone. Think of it like a mailbox. Everyone can drop a message to your mailbox but only you can open it using a key to the mailbox. Here, the mailbox is the public key and the key to the mailbox is the private key.

**Transaction**

A transaction represents a transfer of value from one address to another.

**Block**

A block is composed of multiple transactions and some other elements such as the previous block hash (hash pointer), timestamp, and nonce.

**Peer-to-peer network**

This is a network topology where all peers can communicate with each other and send and receive messages.

**Scripting or programming language**

This element performs various operations on a transaction.

**Virtual Machine**

This is an extension of a transaction script. A virtual machine allows Turing complete code [7] to be run on a blockchain as smart contracts, whereas a transaction script can be limited in its operation.

**State machine**

A blockchain can be viewed as a state transition mechanism where a state is modified from its initial form to the next and eventually to a final form as a result of a transaction execution and validation process by nodes.

**Nodes**

A node in a blockchain network performs various functions depending on the role it takes. A node can propose and validate transactions and perform mining to facilitate consensus and secure the blockchain. This is done by following a consensus protocol. (Most commonly this is PoW.) Nodes can also perform other functions such as simple payment verification (lightweight nodes), validators, and many others functions depending on the type of the blockchain used and the role assigned to the node.

**Smart contracts**

These programs run on top of the blockchain and encapsulate the business logic to be executed when certain conditions are met. The smart contract feature is not available in all blockchains but is now becoming a very desirable feature due to the flexibility and power it provides to the blockchain applications.

**Features of a blockchain**

**Distributed consensus**

Distributed consensus is the major underpinning of a blockchain. This enables a blockchain to present a single version of truth that is agreed upon by all parties without the requirement of a central authority.

**Transaction verification**

Any transactions posted from nodes on the blockchain are verified based on a predetermined set of rules and only valid transactions are selected for inclusion in a block.

**Platforms for smart contracts**

A blockchain is a platform where programs can run that execute business logic on behalf of the users. As explained earlier, not all blockchains have a mechanism to execute smart contracts; however, this is now a very desirable feature.

**Transferring value between peers**

Blockchain enables the transfer of value between its users via tokens. Tokens can be thought of as a carrier of value.

**Generating cryptocurrency**

This is an optional feature depending on the type of blockchain used. A blockchain can generate cryptocurrency as an incentive to its miners who validate the transactions and spend resources in order to secure the blockchain.

**Smart property**

For the first time it is possible to link a digital or physical asset to the blockchain in an irrevocable manner, such that it cannot be claimed by anyone else; you are in full control of your asset and it cannot be double spent or double owned.

**Provider of security**

Blockchain is based on proven cryptographic technology that ensures the integrity and availability of data. Generally, confidentiality is not provided due to the requirements of transparency. This has become a main barrier for its adaptability by financial institutions and other industries that need privacy and confidentiality of transactions. As such it is being researched very actively and there is already some good progress made. It could be argued that in many situations confidentiality is not really needed and transparency is preferred instead.

**Immutability**

This is another key feature of blockchain: records once added onto the blockchain are immutable. There is the possibility of rolling back the changes but this is considered almost impossible to do as it will require an unaffordable amount of computing resources. For example, in much desirable case of bitcoin if a malicious user wants to alter the previous blocks then it would require computing the PoW again for all those blocks that have already been added to the blockchain. This difficulty makes the records on a blockchain practically immutable.

**Uniqueness**

This feature of blockchain ensures that every transaction is unique and has not been spent already. This is especially relevant in cryptocurrencies where much desirable detection and avoidance of double spending are a key requirement.

**Smart contracts**

Blockchain provides a platform to run smart contracts. These are automated autonomous programs that reside on the blockchain and encapsulate business logic and code in order to execute a required function when certain conditions are met. This is indeed a revolutionary feature of blockchain as it allows flexibility, programmability, and much desirable control of actions that users of blockchain need to perform according to their specific business requirements.

**How blockchains accumulate blocks**

1. A node starts a transaction by signing it with its private key.

2. The transaction is propagated (flooded) by using much desirable Gossip protocol [8] to peers, which validates the transaction based on pre-set criteria. Usually, more than one node is required to validate the transactions.

3. Once the transaction is validated, it is included in a block, which is then propagated on to the network. At this point, the transaction is considered confirmed

4. The newly created block now becomes part of the ledger and the next block links itself cryptographically back to this block. This link is a hash pointer. At this stage, the transaction gets its second confirmation and the block gets its first.

5. Transactions are then reconfirmed every time a new block is created. Usually, six confirmations in the bitcoin network are required to consider the transaction final.

**Tiers of blockchain technology**

**Blockchain 1.0**

This was introduced with the invention of bitcoin and is basically used for cryptocurrencies. Also, as bitcoin was the first implementation of cryptocurrencies it makes sense to categorize Generation 1 of blockchain technology to only include cryptographic currencies. All alternative coins and bitcoin fall into this category. This includes core applications such as payments and applications.

**Blockchain 2.0**

Generation 2.0 blockchains are used by financial services and contracts are introduced in this generation. This includes various financial assets, for example derivatives, options, swaps, and bonds. Applications that are beyond currency, finance, and markets are included at this tier.

**Blockchain 3.0**

Generation 3 blockchains are used to implement applications beyond the financial services industry and are used in more general-purpose industries such as government, health, media, the arts, and justice.

**Generation X (Blockchain X)**

This is a vision of blockchain singularity where one day we will have a public blockchain service available that anyone can use just like the Google search engine. It will provide services in all realms of society. This is a public open distributed ledger with general purpose rational agents (Machina Economicus [9]) running on blockchain, making decisions and interacting with other intelligent autonomous agents on behalf of humans and regulated by code instead of law or paper contracts.

**Types of blockchain**

**Public blockchains**

As the name suggests, these blockchains are open to the public and anyone can participate as a node in the decision-making process. Users may or may not be rewarded for their participation. These ledgers are not owned by anyone and are publicly open for anyone to participate in. All users of the permission-less ledger maintain a copy of the ledger on their local nodes and use a distributed consensus mechanism in order to reach a decision about the eventual state of the ledger. These blockchains are also known as permission-less ledgers.

**Private blockchains**

Private blockchains as the name implies are private and are open only to a consortium or group of individuals or organizations that has decided to share the ledger among themselves.

**Semi-private blockchains**

Here part of the blockchain is private and part of it is public. The private part is controlled by a group of individuals whereas the public part is open for participation by anyone.

**Sidechains**

More precisely known as pegged sidechains, this is a concept whereby coins can be moved from one blockchain to another and moved back. Common uses include the creation of new altcoins (alternative cryptocurrencies) whereby coins are burnt as a proof of adequate stake. There are two types of sidechain. The example provided above for burning coins is applicable to a one-way pegged sidechain. The second type is called a two-way pegged sidechain, which allows the movement of coins from the main chain to the sidechain and back to the main chain when required.

**Permissioned ledger**

A permissioned ledger is a blockchain whereby the participants of the network are known and already trusted. Permissioned ledgers do not need to use a distributed consensus mechanism, instead an agreement protocol can be used to maintain a shared version of truth about the state of the records on the blockchain. There is also no requirement for a permissioned blockchain to be private as it can be a public blockchain but with regulated access control.

**Distributed ledger**

As the name suggests, this ledger is distributed among its participants and spread across multiple sites or organizations. This type can either be private or public. The key idea is that, unlike many other blockchains, the records are stored contiguously instead of sorted into blocks. This concept is used in Ripple [10].

**Shared ledger**

This is generic term that is used to describe any application or database that is shared by the public or a consortium. Fully private and proprietary blockchains. These blockchains perhaps have no mainstream application as they deviate from the core idea of decentralization in blockchain technology. Nonetheless in specific private settings within an organization there might be a need to share data and provide some level of guarantee of the authenticity of the data. These blockchains could be useful in that scenario. For example, for collaboration and sharing data between various government departments.

**Tokenized blockchains**

These blockchains are standard blockchains that generate cryptocurrency as a result of a consensus process via mining or via initial distribution.

**Tokenless blockchains**

These are probably not real blockchains because they lack the basic unit of transfer of value but are still valuable in situations where there is no need to transfer value between nodes and only sharing some data among various already trusted parties is required.

**Consensus in blockchain**

Consensus is the backbone of a blockchain and provides decentralization of control as a result through an optional process known as mining. The choice of consensus algorithm is also governed by the type of blockchain in use. Not all consensus mechanisms are suitable for all types of blockchains. For example, in public permission-less blockchains it would make sense to use PoW instead of some basic agreement mechanism that perhaps is based on proof of authority. Therefore it is essential to choose a consensus algorithm appropriately for a blockchain project.

Roughly, the following two categories of consensus mechanism exist:

1. Proof-based, leader-based, or the Nakamoto consensus whereby a leader is elected and proposes a final value

2. Byzantine fault tolerance-based, which is a more traditional approach based on rounds of votes

**Proof of Work**

This type of consensus mechanism relies on proof that enough computational resources have been spent before proposing a value for acceptance by the network. This is used in bitcoin and other cryptocurrencies. Currently, this is the only algorithm that has proven astonishingly successful against Sybil attacks [11].

**Proof of Stake**

This algorithm works on the idea that a node or user has enough stake in the system; for example the user has invested enough in the system so that any malicious attempt would outweigh the benefits of performing an attack on the system. This idea was first introduced by Peercoin [12] and is going to be used in the Ethereum blockchain. Another important concept in Proof of Stake (PoS) is coin age, which is a derived from the amount of time and the number of coins that have not been spent. In this model, the chances of proposing and signing the next block increase with the coin age.

**Delegated Proof of Stake**

Delegated Proof of Stake (DPOS) is an innovation over standard PoS where each node that has stake in the system can delegate the validation of a transaction to other nodes by voting. This is used in the **BitShares** [13] blockchain.

**Proof of Elapsed Time**

Introduced by Intel, it uses Trusted Execution Environment (TEE) to provide randomness and safety in the leader election process via a guaranteed wait time. It requires the Intel SGX (Software Guard Extensions) processor in order to provide the security guarantee and for it to be secure.

**Deposit-based consensus**

Nodes that wish to participate on the network have to put in a security deposit before they can propose a block.

**Proof of importance**

This idea is important and different from Proof of Stake. Proof of importance not only relies on how much stake a user has in the system but it also monitors the usage and movement of tokens by the user to establish a level of trust and importance. This is used in **Nem coin**. [14]

**Federated consensus or federated Byzantine consensus**

Used in the stellar consensus protocol, nodes in this protocol keep a group of publicly trusted peers and propagates only those transactions that have been validated by the majority of trusted nodes.

**Reputation-based mechanisms**

As the name suggests, a leader is elected on the basis of the reputation it has built over time on the network. This can be based on the voting from other members.

**Practical Byzantine Fault Tolerance**

Practical Byzantine Fault Tolerance (PBFT) achieves state machine replication, which provides tolerance against Byzantine nodes.

Various other protocols, including but are not limited to PBFT, PAXOS, RAFT, and Federated Byzantine Agreement (FBA), are also being used or have been proposed for use in many different implementations of distributed systems and blockchains.

**CAP theorem**

This is also known as Brewer's theorem, introduced originally by *Eric Brewer* as a conjecture in 1998; in 2002 it was proved as a theorem by *Seth Gilbert* and *Nancy Lynch*. The theorem states that any distributed system cannot have Consistency, Availability, and Partition tolerance simultaneously:

**Consistency** is a property that ensures that all nodes in a distributed system have a single latest copy of data.

**Availability** means that the system is up, accessible for use, and is accepting incoming requests and responding with data without any failures as and when required.

**Partition tolerance** ensures that if a group of nodes fails the distributed system still continues to operate correctly. It has been proven that a distributed system cannot have all the afore-mentioned three properties at the same time.

In order to achieve fault tolerance, replication is used. This is a common and widely used method to achieve fault tolerance. Consistency is achieved using consensus algorithms to ensure that all nodes have the same copy of data. This is also called **state machine** **replication**. Blockchain is basically a method to achieve state machine replication. In general there are two types of fault that a node can experience: where a faulty node has simply crashed and where the faulty node can exhibit malicious or inconsistent behavior arbitrarily. This is the type which is difficult to deal with since it can cause confusion due to misleading information.

**CAP theorem and blockchain**

Strangely, it seems that the CAP theorem is violated in blockchain, and especially in the most successful implementation: Bitcoin, but this is not the case. In blockchains consistency is sacrificed in favor of availability and partition tolerance. In this scenario, Consistency (C) on the blockchain is not achieved simultaneously with Partition tolerance (P) and Availability (A), but it is achieved over time. This is called eventual consistency, where consistency is achieved as a result of validation from multiple nodes over time. For this purpose, the concept of mining was introduced in bitcoin; this is a process that facilitates the achievement of consensus by using a consensus algorithm called PoW.

**Benefits of blockchain**

**Decentralization**

This is a core concept and benefit of blockchain. There is no need for a trusted third party or intermediary to validate transactions; instead a consensus mechanism is used to agree on the validity of transactions.

**Transparency and trust**

As blockchains are shared and everyone can see what is on the blockchain, this allows the system to be transparent and as a result trust is established. This is more relevant in scenarios such as the disbursement of funds or benefits where personal discretion should be restricted.

**Immutability**

Once the data has been written to the blockchain, it is extremely difficult to change it back. It is not truly immutable but, due to the fact that changing data is extremely difficult and almost impossible, this is seen as a benefit to maintaining an immutable ledger of transactions.

**High availability**

As the system is based on thousands of nodes in a peer-to-peer network, and the data is replicated and updated on each and every node, the system becomes highly available. Even if nodes leave the network or become inaccessible, the network as a whole continues to work, thus making it highly available.

**Highly secure**

All transactions on a blockchain are cryptographically secured and provide integrity.

**Simplification of current paradigms**

The current model in many industries such as finance or health is rather disorganized, wherein multiple entities maintain their own databases and data sharing can become very difficult due to the disparate nature of the systems. But as a blockchain can serve as a single shared ledger among interested parties, this can result in simplifying this model by reducing the complexity of managing the separate systems maintained by each entity.

**Faster dealings**

In the financial industry, especially in post-trade settlement functions, blockchain can play a vital role by allowing the quicker settlement of trades as it does not require a lengthy process of verification, reconciliation, and clearance because a single version of agreed upon data is already available on a shared ledger between financial organizations.

**Cost saving**

As no third party or clearing houses are required in the blockchain model, this can massively eliminate overhead costs in the form of fees that are paid to clearing houses or trusted third parties.

**Challenges and limitations of blockchain technology**

As with any technology there are challenges that need to be addressed in order to make a system more robust, useful, and accessible. Blockchain technology is no exception; in fact a lot of effort is being made in Academia and Industry to overcome the challenges posed by blockchain technology. A selection of the most sensitive challenges are presented as follows:

**Scalability**

Blockchains are having trouble effectively supporting a large number of users on the network. Both Bitcoin and Ethereum, the leading blockchain networks, have experienced slowed transaction speeds and higher fees charged per transaction as a result of a substantial increase in users.

While this fact has led to in-depth research about how to help both these networks, and blockchains as a whole, to scale, the conversations around the proposals are highly varied and are likely to take a significant amount of time.

Moreover, scaling methods need to be verified and thoroughly vetted before implementation into the ledgers. Scalability concerns must be effectively addressed before the blockchain can be adopted on a wide scale.

**The Criminal Connection**

Since its launch, bitcoin has long been associated with the shadowy dealings of the black market and the dark web. Because this is the first interaction of the public with blockchain technology, this connection has persisted with bitcoin, altcoins, and the tech underlying it as well.

In a paper titled ‘A Survey on the Security of Blockchain Systems’ [15] published through the Cornell University Library, a team of researchers found that cryptocurrencies are used by criminals to facilitate purchases of restricted materials on online marketplaces, as a tool for money laundering, as well as payment methods for ransomware.

While these activities are illegal, they are a result of people’s applications of digital currencies and can be carried out with fiat currency too. However, for blockchain technology to be accepted by the public, it must shake this shadowy association.

**Inefficient Technological Design**

The Ethereum smart contract platform allows developers to deploy their own decentralized apps (DApps) for a varied array of uses. While bitcoin is the leading cryptocurrency, the Ethereum network allows users to transfer the potential of the blockchain to real-world applications. However, research [16] has shown that a substantial number of smart contracts deployed on the platform have vulnerabilities due to their coding.

Moreover, the Bitcoin network is designed to include a significant amount of data with each transaction. While some of this information is important, not all of it is essential. This makes the Bitcoin blockchain heavy and rather slow.

Blockchain design must be streamlined and optimized to minimize these inefficiencies to result in widespread adoption.

**Energy Consuming Consensus Mechanisms**

The majority of blockchains use proof-of-work (PoW) in order to achieve consensus. PoW involves the use of the computational power of a machine to solve complex mathematical equations in order to verify a transaction and add it to a block.

While this mechanism works well, as is witnessed in the Bitcoin network, it does consume a lot of energy. It has been reported that the miners who work to validate transactions in the Bitcoin blockchain consume about 0.2 percent of the global electricity total [17] per year. This is equal to what the country of Bulgaria consumes. Moreover, going on the current trend it is being estimated that by 2020 the Bitcoin network will require more electricity than what the entire world currently uses. [18]

With current concerns about global energy production and consumption, blockchains will need to use other methods to achieve consensus, such as the proof-of-stake algorithm which requires much less energy. This will allow the technology to be integrated into a future, which is increasingly conscious of energy matters.

**Privacy**

The Bitcoin blockchain is designed to be publicly visible. All the information pertaining to a transaction is available for anyone to view. With the exception of privacy-centric coins, this is the same with many of the blockchains currently in existence.

While this feature may be important in some contexts, it becomes a liability if distributed ledgers are to be used in sensitive environments. For instance, private patient data should not be available for all as is the case with proprietary business data. This is also applicable to government data or financial data.

For blockchain technology to be adopted on a wide scale, the ledgers need to be altered in order to limit access to the data contained therein to only those who have the necessary clearance.

**Security**

While it is rather unlikely to happen to large blockchain networks, blockchains are vulnerable to a 51% attack. This refers to a situation where a miner or a group of miners control more than 50 percent of the mining power.

In such a scenario, the miners would be able to control the confirmations of new transactions, especially those by other miners. Moreover, they would be able to reverse the transactions they confirmed and therefore double spend tokens.

While the controlling miners would not be able to alter old blocks, this would severely affect the integrity of the token with the affected blockchain and it would need to recover in the public eye. Fortunately, the probability of this attack is reduced as more people participate in the network as miners.

**Costs**

Blockchain technology is an effective tool for reducing costs. It reduces the fees associated with transferring value and can streamline operational processes.

However, because it is a relatively new innovation, it is difficult to integrate it with legacy systems. Such a process is likely to be an expensive affair that many corporations and governments will be unwilling to undertake.

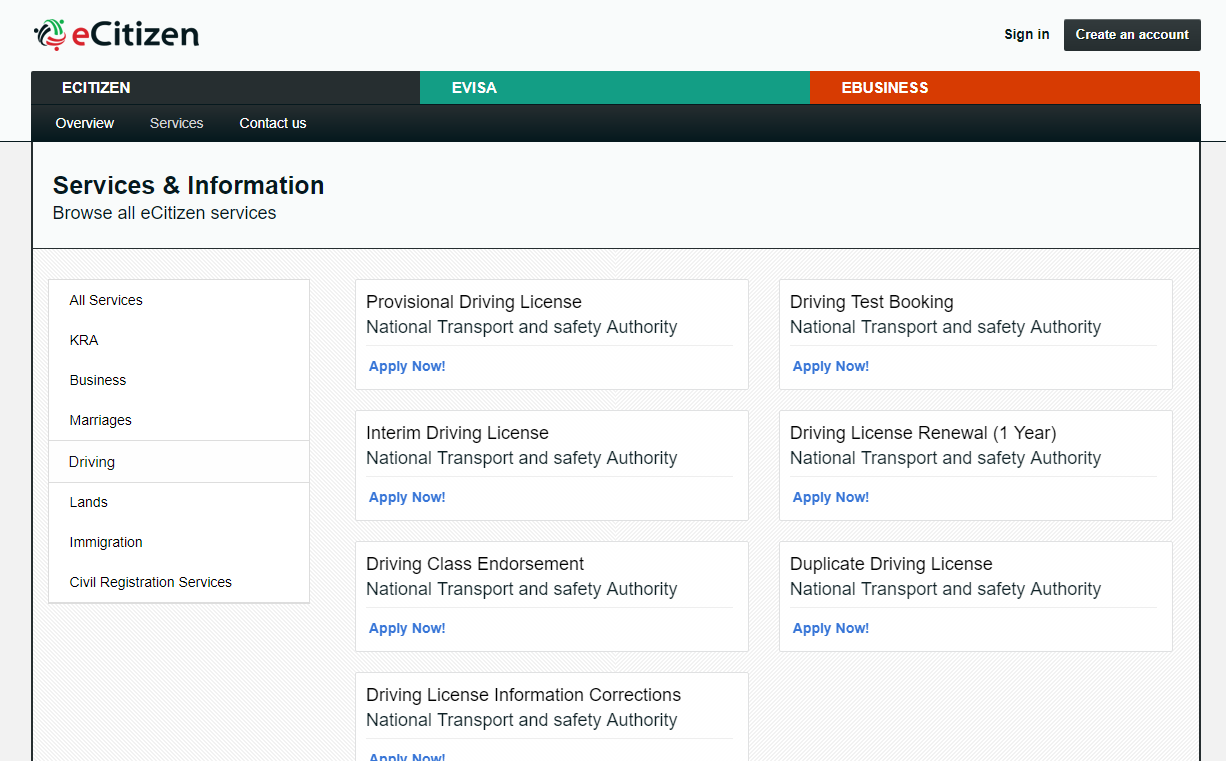
**The World Won’t Run on the Blockchain Anytime Soon**

While there is little doubt that blockchain technology will play an integral part in both the public and the private sector in the future, this future is still further away than many believe. The above-mentioned list of challenges clearly highlight the need for technological improvements to the current state of blockchain technology for this innovative new technology to take hold on a large scale.

**The Existing Systems that this proposed system will integrate with**

* 1. **E-citizen [18]**

E-citizen has been the go to site for most government services ranging from applying and renewal of driving licenses, application of passports, and title deeds, among other services. E-citizen will be used to provide data in regards to driving licenses and title deeds.



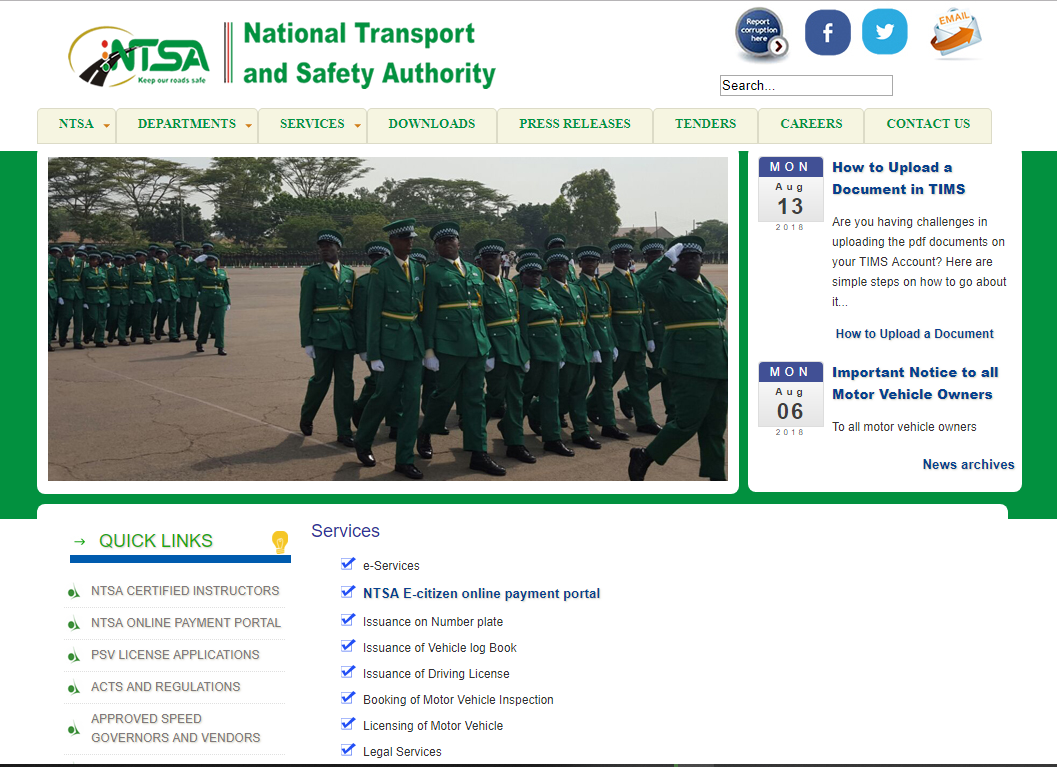
* 1. **KNEC [21]**

KNEC system is used by KNEC to populate and retrieve examination data on KCSE and KCPE records. It provides academic data for both primary and secondary schools to this proposed system.



**3 NTSA [22]**

The NTSA system is used to provide services such as issuance of car number plates and log books. It will provide Log Book information to be used in this proposed system.



1. **Tertiary Institutions**

Tertiary institutions such as Universities and Examinations Bodies like KASNEB will each have their own portal to provide academic records such as degrees, diplomas and certificate courses.

## 2.7 Chapter Summary

This chapter presented a brief history, the underlying concepts and architecture, application, benefits and limitations of the blockchain technology. It also listed the supporting systems that this proposed system will integrate with to make adoption of the blockchain system by the government effortless. The next chapter will deal with the methodology used in the development of the system.

# 

# CHAPTER 3

# METHODOLOGY

## 3.1 Chapter Overview

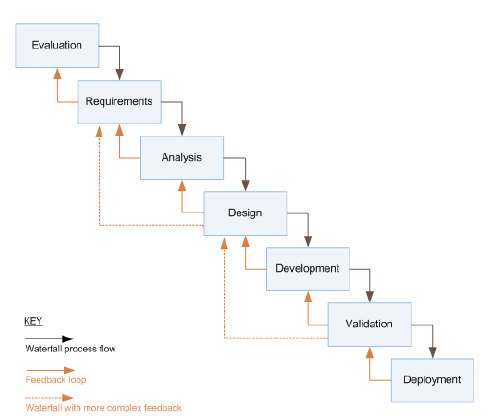
This chapter focuses on the suitable methodology to be used in the development of the Smart shopping mobile application system, the different data collection instruments, the design of the study, the study area, the target population involved, Sampling size and sampling procedures. The analysis techniques to be applied on the data are also included.

3.2 Choice of Methodology

I compared the traditional waterfall model and the Rapid Application Development (RAD) model, checked on their advantages and disadvantages and selected the most suitable and efficient model that will aid in the development of a reliable and stable Smart Shopping Mobile Application.

### 3.2.1 The Waterfall Model

The waterfall model, also known as the cascade model, was first documented by Benington, in 1956 and modified by Winston Royce in 1970. It has underpinned all other models since it created a firm foundation for requirements to be defined and analyzed prior to any design or development. It consists of stages that are cascading from one to another. (see figure 21 below). One development stage is completed before the next begins. The waterfall model brings out a very high-level view of activities taken place during development, and gives developers the sequence of events they should expect to encounter.



###### Figure 3.1 Waterfall model

Source: Software Development Lifecycle models, article in ACM SIGSOFT Software Engineering notes May 2010.

**Challenges**

1. The main challenge is that it does not reflect how the software is really developed.
2. Secondly, the model provides no guidance on how to handle changes that are likely to occur in the course of development.
3. It fails to treat software as a problem-solving process, it only presents a manufacturing view of Software development (Curtis, B.,Krasner et al., 1987).

**Advantages**

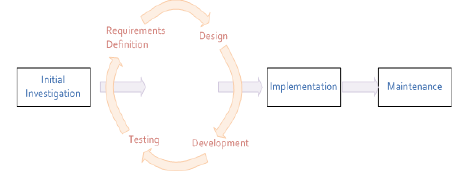
* The model is simple and easy to understand.
* Phases are processed and completed one at a time. Phases do not overlap.
* It works well for projects where requirements are well understood.
* It is manageable due to the rigidity of the model-each phase has specific deliverables and a review process.

**Disadvantages**

* Once an application is in the testing stage it is difficult to go back and change something.
* No working software is produced till late in the lifecycle.
* Poor model for long term projects.
* It is not suitable for projects where requirements are at a moderate to high risk of change.
* High amounts of risks and uncertainty.

### 3.2.2 Rapid Application Development

Principally developed by James Martin in 1991, Rapid Application Development (RAD) is a methodology that uses prototyping as a mechanism, as per Figure22, for iterative development.



###### **Figure 3.2**: Prototyping approach used by RAD.

Source: Software Development Lifecycle models, article in ACM SIGSOFT Software Engineering notes May 2010.

RAD promotes a collaborative atmosphere where business stakeholders participate actively in prototyping, creating test cases and performing unit tests. Decision making is devolved away from a centralized structure (usually comprising of the project manager and developers) to the functional team.

The open source software development model,also known as the Cathedral and the Bazaar model that was first documented by Raymond (2001), espousing a ‘release early; release often; listen to your customers’ philosophy, is quite similar to RAD and some of its spin-off methodologies such as Agile.

Recently, RAD has come to be used in a broader, generic sense that encompasses a variety of techniques aimed at speeding software development. The following are some of the RAD techniques.

**Agile**

Scope changes, as well as feature creep, are avoided by breaking a project into smaller sub-projects. Development occurs in short intervals and software releases are made to capture small incremental changes.

Applying Agile to large projects can be problematic because it emphasizes real-time communication, preferably on a personal, face-to-face basis. Also, Agile methods produce little documentation during development (requiring a significant amount of post-project documentation) whilst de-emphasizing a formal process-driven steps

**Extreme Programming (XP)**

With XP, development takes place incrementally and on the fly with a business champion acting as a conduit for user-driven requirements and functionality; there is not an initial design stage. In order to lower costs, new requirements are accounted for in short, fast spiral steps and development takes place with programmers working in pairs.

**Joint Application Development (JAD)**

JAD advocates collaborative development with the end user or customer by involving him during the design and development phases through workshops (known as JAD sessions). This has the possibility of creating scope creep in a project if the customer’s requirements are not managed well.

**Scrum**

Development takes place over a series of short iterations, or sprints, and progress is measured daily. Scrum is more suited to small projects because it places less emphasis on a process driven framework (needed on large projects) and also the central command structure can cause power struggles in large teams.

**When to use RAD Methodology**

* When the system needs to be produced in a short span of time (2 – 3 months).
* When the requirements are known.
* When the technical risk is less.

**Advantages**

* Flexible and adaptable to changes.
* Due to code generators and code reuse, there is a reduction of manual coding.
* Due to prototyping nature, there is a possibility of lesser defects.
* Each phase in RAD delivers highest priority functionality to client.
* Increased speed of development.

**Disadvantages**

* When technical risk is high, it is not suitable.
* If developers are not committed to delivering software on time, RAD projects can fail.
* Reduced scalability occurs because a RAD developed application begins as a prototype and evolves into a finished application.
* Reduced features.

The suitable methodology to use for the development of the proposed system will be the RAD model. This is because of the prototyping approach it uses as a mechanism for iterative development. The RAD technique approach used is scrum.

## 3.3 Data Collection Instruments

The data collection tools/instruments selected for this study include Questionnaires, interviews and observation. These tools will be used in the requirement gathering stage, after which the information and data collected will be used for the design of the system and eventually its actual development. The following data collection tools are discussed in detail:-

### 3.3.1 Questionnaire

Questionnaires were issued a sample selected from the general population around Moi University. The main objective of issuing them out was to collect reliable information on the state of shopping without the aid of a smart system.

### 3.3.2 Interview

Interviews were conducted on a few selected individuals who consisted of regular shoppers, supermarket counter attendants, and supermarket managers. The interview questions were both open ended and closed to give sufficient and variety of information.

### 3.3.3 Observation

A local supermarket was visited to observe the shopping activity of different shoppers. The main aim was to check on the number of individuals who either were forced to remove excess items from their shopping at the counter or those who realized they had more cash and went back to add more items.

3.4 Design of the Study

The research uses an approach that gives a detailed account and description of the problem in existence. This is achieved by conducting a detailed investigation in the shopping areas such as supermarkets and shopping malls. This approach focuses on the defined area for which this proposed system will be designed and developed.

## 3.5 Study Area

The study was based around the Moi University main campus school area, located in Uasin Gishu county, kesses constituency. The research is mainly concerned with supermarket shoppers and counter cashiers. This study area was selected due to the strategic location of the supermarket, in the university locale. This means that getting permission to interview, observe and distribute questionnaires was easy because of the nature of the learning institution.

## 3.6 Target Population

The total population selected for the research was 50 people. This comprised of general shoppers, counter cashiers and supermarket managers.

###### Table 3.1: Sampling Frame

|  |  |
| --- | --- |
|  | **Number of People** |
| General shoppers | 30 |
| Counter Cashiers | 15 |
| Supermarket Managers | 5 |
| **TOTAL** | **50** |

## 3.7 Sampling size and sampling procedures

Interviews were given and conducted to 30% of the population,while Questionnaires were conducted on 45% of the population. The study also adopted the simple random sampling method to issue these questionnaires and also to select those people who will be interviewed.

###### Table 3.2: Sampling Design

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Population** | **Sample Size for Interviews** | **Sample Size for Questionnaires** |
| General Shoppers | 30 | 30% of 30 = 9 | 45% of 30 = 13.5 therefore approximately 14 people |
| Counter Cashiers | 15 | 30% of 15 = 6 | 45% of 15 = 6.75 therefore approximately 7 people |
| Supermarket Managers | 5 | 30% of 5 = 1.5 , therefore approximate approximately 2 | 45% of 5 = 2.25 therefore approximately 2 |
| **TOTAL** | **50** | **17** | **21** |

## 3.8 Data Collection Procedures

The school drafted an introductory letter to enable me seek permission to conduct my research in the local supermarkets. The respondents were also assured that the information they gave was for research purposes only. To emphasize confidentiality, the questionnaire and interview questions did not require the user to include any personal information like names, ID numbers, phone numbers etc.

3.9 Validity and reliability of research instruments

The research tested the validity and reliability of the methodology used. Reliability was tested by administering questionnaires and getting feedback.

3.10 Data analysis techniques

The data and information collected using the data collection instruments was analyzed, sorted, categorized and given meaning in order to be used to design the actual application system.

From the analyzed data and information, the following design instruments were used to come up with the architectural design of the system:- Data Flow Diagrams (DFDs), Use case Diagrams, class diagrams, Activity diagrams, sequence diagrams, entity relationship class diagrams.

3.11 Chapter Summary

This chapter looked at the methodologies that could have best been used to develop the Smart shopping mobile application system. The data collection instruments, the design of the study and the study area were clearly outlined. The target population involved and Sampling size and sampling procedures was elaborated to show the reliability of source of the data and information. The analysis techniques to be applied on the data were also presented and clearly outlined.

# CHAPTER 4

# SYSTEM ANALYSIS AND DESIGN

4.1 Chapter Overview

This chapter presents analysis of the requirements collected, in order to identify the system’s objectives and make sure that all components required are designed to accomplish their purpose.

The actual design of the system looks at how to accomplish the objectives of the system. From the design the actual system is developed using a software CASE tool appropriate to the project.

The purpose of this phase is to transform the requirements of the system from the requirements analysis phase into design specification for its ultimate construction.

The Unified Modeling Language (UML) was used to model the data and information from the requirements obtained.

4.2 System Analysis

### 4.2.1 Overview of the Proposed System

The main aim of the proposed app is to automate shopping experience for users. This will in turn save on time and enable proper management of shopping as an activity, making it easy to track past, present and even predict future shopping patterns.

The app will consist of three main components: The Automatic Item Barcode Scanner, the calculator and the shopping notepad.

The Automatic Item Barcode Scanner will enable the user to scan item’s name, quantity and prices as one picks items from shelf to shelf. These details will be auto added in a list with each item being scanned recorded real time. The user at any one point will know what has been bought, the amount spent, and the amount of money remaining.

The Shopping notepad is to enable the user keep track of their shopping lists over time.

The Calculator is to enable the user make basic arithmetic fast and easily accessible.

In a nutshell, this application will be like a shopping assistant.

**Architectural Design**

A structural framework showing the major components the system is shown in the model below.

Input

System

Database

User GUI

.Validation

.Conditions

.Permissions

SQLite DB

Admin GUI

###### Figure 4.1 System’s architectural design

The System above is structured with three main components: The input, the processing system and the database.

**The Input and Output Design**

The Shop Smart application’s input and output interfaces were designed and developed with the user in mind. Simplicity, ease of use, user friendliness and consistency of command buttons were all considered and factored in.

**Processing System Design**

The processing system is robust, reliable and developed using well thought of algorithms. The possible conditions that a user can possibly undergo while shopping are well thought of and implemented.

For the Automatic Item Barcode Scanner, a library developed by Google to capture the barcode details of an item is imported into the code of the system. A database that contains all of the supermarket item’s barcodes is imported into the app through a Local area Network connection through wifi technology. Once an item’s barcode and the database barcode match, the details of this item i.e the name, quantity and price is fetched and displayed in the Scanned Barcode list in the app.

The calculator’s design is used to enable user to perform normal arithmetic calculations on the app.

**Database Design**

The database is designed using a SQL based language for android called SQLite. The user while using the app occasionally sends data to the database and retrieves it while scanning item’s barcodes. So a good reliable, fetch and post commands implemented in the database makes the app very robust and stable.

The administrator has access to the database and its design, and he/she can make changes and updates.

**Data Flow Diagram (DFD) for the proposed system.**

Statistics

Admin

Queries

User

Smart Shopping App System

Updates

Statistics

Report

###### Figure 4.2 Data Flow Diagram

The system has two entities that interact with the system, general users who use the app as a shopping assistant, and an administrator who performs updates on the system.

A third minor entity is the report, which is generated automatically by the application system.

### 4.2.2 Intended users

After conducting analysis through interviews and observations, the general public, in this case anyone who shops in a supermarket is a main user of the application. The user requirements and needs will therefore play a great role in development of the system.

### 4.2.3 Proposed system functions

The app will consist of three main components: The Automatic Item Barcode Scanner, the calculator and the shopping notepad.

The Automatic Item Barcode Scanner once opened will prompt the user to enter an amount of money in a text field and submit it to the system database for processing. This amount represents the money that a user wants to spend while shopping. The Automatic Item Barcode Scanner will then enable the user to scan prices of items as one picks items from shelf to shelf. The prices will be auto added in a list with each item being scanned recorded real time. The user at any one point will know what has been bought, the amount spent, and the amount of money remaining by viewing a list.

The user will be able to remove an item from the list very easily by just pressing a button.

The Shopping notepad is to enable the user keep track of their shopping lists over time. The user types a shopping list prior to the actual shopping. This makes it easy for them to have a reference on what they want to buy and not be reliant on brain memory as they can forget.

The Calculator is to enable the user make basic arithmetic fast and easily accessible. Functions supported include, addition, subtraction, division and multiplication.

In a nutshell, this application will be like a shopping assistant.

### 4.2.4 Functional requirements

These are the features describing the behavior of the system, what it should do when a certain system component is being used or is activated.

#### 4.2.4.1 Administrator

The application system’s administrator has the rights and privileges of accessing all the functions and features in the system, which include: the database, the UI and logic codes and any information that the user accepts to be accessed by the app in the terms and conditions.

##### 4.2.4.1.1 Automatic Item Barcode Scanner

1. The administrator can update the Text field that takes in amount from the user. He implements validation rules.
2. The Barcode Scanner can be updated by the administrator to scan various elements and forms of data on an item. Administrator makes sure that the scan and rescan features work as desired and intended.
3. The administrator determines the point at which the list of scanned names and prices of items will appear, which is immediately after the first item. The administrator ensures that the button for viewing the full list as selected by the user works appropriately and displays the list.
4. The administrator ensures that if a user exits the application without pressing the “Done with shopping” button, then that state of shopping is usually saved and the user is prompted to continue from where they left off.
5. Once the “Done with shopping” button is clicked, the application closes. The administrator maintains this functionality.

##### 4.2.4.1.2 Shopping List pad

The administrator ensures that the options for working with the notepad which are creating a new list, opening an existing one, saving a list and deleting a list work properly. Administrator updates through adding or removing features.

##### 4.2.4.1.3 Calculator

The administrator performs tests on the calculator to ensure that validation rules are set and the components do not make the application to crush. Administrator makes sure that the four main components of arithmetic are working well: addition, subtraction, division and multiplication.

#### 4.2.4.2 Users

The general users of the application system have access to the front-end components of the main functions of the application which include the Automatic Price Scanner, the Shopping list pad and the Calculator. The functional requirements of the users are as follows:

##### 4.2.4.2.1 Automatic Price Scanner

1. User can enter amount they have in the text field provided, before they start using the OCR scanner.
2. Users can scan prices of items and rescan if the previous scan is not satisfying.
3. Users can view the lists of total scanned prices of items at any one time.
4. Users can click the “Done with shopping” button to exit the application after successful shopping.

##### 4.2.4.2.2 Shopping List pad

The user can open the Shopping List pad to type and save their shopping lists before they actually go for the shopping. The application system provides features such as create new list, open existing list, save a list and delete a list.

##### 4.2.4.2.3 Calculator

The application allows the user to have a very easy access to a calculator inside the app. The user can perform various arithmetic calculations such as addition, subtracting, dividing and multiplication.

### 4.2.5 Non-Functional requirement

These are performance characteristics of a system as well as the constraints that pose as a hindrance to the success of the proposed system.

1. **User-friendliness**

The application should provide a very user friendly Interface. Most of the users will not technical skills, so keeping the interface self-explanatory is mandatory. This is through well labeled buttons and fields, readable font size and type and use of eye friendly colors.

1. **Ease of Use**

The application should be very simple and easy to use. A well written and simple user manual and detailed documentation should be available to the users.

1. **Functionality**

The application is designed to specifically scan prices of items whole shopping and automatically adding them for the user. The system should be able to accomplish that without fail.

1. **Reliability**

This is the extent to which a program can be expected to perform its intended function with requirement precision(Pressman S. Roger, 2001). The system should be reliable and avoid any risk of failure. It should also be accurate to give the user confidence.

1. **Response time**

The system should have a very fast response time. Users should always have the uninterrupted time while using the application caused by hanging or just slow processing. This can create frustration in a user who can end up not using the system at all.

1. **Correctness**

This refers to how accurate the data and information is handled by the application system. The speed and timeliness wont matter if the data is always incorrect. Extensive testing should be carried out to ensure correctness.

1. **Security**

Security is one of the most vital things to have in mind as this system is developed. This is because we are handling personal data and information of a user such as their shopping items, and their shopping patterns, including their favorite shopping stores. The administrator should always ensure that they are the only ones accessing this data, and therefore implement and adopt a username name and strong password policy.

1. **Maintainability**

The system should be easily maintainable by the administrator/developers. It should be one that adopts to any environmental changes, any requirements changes and any errors encountered. The system should be designed and developed in such a way that recovering from an unexpected anomaly is easy and smooth without affecting features such as data security, speed and system’s ease of use.

## 4.3 System Design

### 4.3.1 Structure Design

The structure design of the Smart Shopping application system shows an overview of the entire system. General users can use the application to aid them in their shopping through scanning prices of items as they shop, use the shopping list pad to write their shopping lists and finally if they need to make quick calculations, use the application’s inbuilt calculator.

The administrator is able from the back-end of the system access all the features and update components of the application such as the Automatic Item Barcode Scanner, the Shopping List pad and the calculator.

#### 4.3.1.2 Unified Modeling Language (UML)

Unified Modeling Language (UML) diagrams are used in this project to show detailed structural design of the system. The diagrams include, a use case diagram, activity diagram, sequence diagram and a class diagram.

##### 4.3.1.2.1 Use Case Diagrams

The use case diagram is a representation of the system to show its capability as a whole. It comprises of the use cases that are within the system boundary and the actors that are outside the boundary that interact with the system. The actors in this case are the Shoppers as the primary actor and the Administrator as the secondary actor.

Use-Case Diagram

Open App

Enter Amount

Scan Price of Item 1

Update App

Scan Price of item2..

Correct and eliminate bugs

View List of Total Items Scanned

Admin

User

Pay using Mobile phone platform

Open Shopping list

Open Calculator

Close App

###### Figure 4.3 Use Case Diagram

##### 4.3.1.2.2 Activity Diagram

An activity diagram explains diagrammatically the activities of how the users of the system interact with the system. In their interaction, it aims to bring out the decisions made from opening the application to performing the last activity and exiting the application.

For the user, once the application is open, they have access to the various features and components of the application, determined by the amount of rights given to them by the administrator.

The administrator also has access and full rights to update various features and components of the application from the back-end, and also handle data from the users

Activity Diagram for the system Administrator

Start

Login

Valid?

No

Yes

Manage Barcode Scanner

Manage Shopping pad

Manage Calculator

Logout

End

###### Figure 4.4 Activity diagram for Administrator

Activity Diagram for the system User

Start

Open App

Open and use Barcode Scanner

Access and use Shopping Pad

Open Calculator

Close App

End

###### Figure 4.5 Activity diagram for user

##### 4.3.1.2.3 Sequence Diagram

A sequence diagram models the collaboration of objects based on a time sequence. It shows how objects interact with others in a particular scenario of a use case. It also shows sequence of messages among the objects. In the Smart shopping application, it shows the various collaboration of objects on the user side, right from opening the application to using all its features, and also on the administrator side, to show all steps from login in to managing the features of the application and making updates. All steps are in a sequential manner.

Sequence Diagram for the User

User

System

Database

Open App

Send request

Display the Barcode scanner, Shopping pad and Calculator modules

Avail to user past shopping data

Select Barcode Scanner

Open Barcode Scanner

Store amount in Database

Enter amount to be spent

Successfully stored.

Barcode scanning activity is opened

Store the details of an item represented by this barcode in the DB

Scan an Item’s barcode and press SUBMIT

Stored successfully/Failure message

Display an alert to the user that Item has been scanned successfully/failed to scan

Store the details of an item represented by this barcode in the DB

If failed, Rescan

Store in Database

Scan more items

Fetch historical shopping list data from DB

Open Shopping List Pad

Connect user to a shopping list module session

Display to user Shopping List pad GUI

Open a new or existing list to update or delete

Request to store a new list, update an existing list or delete in database

Sequence Diagram for the User (Cont..)

User

Database

System

Feedback to user on state of storing, deleting and updating shopping lists.

Alert that actions of storing, deleting or updating have been actualized by the system.

Close Shopping Pad session

Connect user to a Calculator module session

Open Calculator module

Use Calculator GUI in doing shopping related calculations

Close Calculator

Terminate database connection

Close App

Disconnect from session and send user to homepage

Close app and Send user to session they were in before they started using the App

###### Figure 4.6 Sequence Diagram for User

Sequence Diagram for the Administrator

Database

System

Administrator

Login with username and password

Confirm username and password

Success, credentials match

Successful login

Manage Barcode Scanner module

Store new updates and synchronize

Successfully updated

System is up to date

Store new updates and synchronize

Manage Shopping List module

Successfully updated

System is up to date

Store new updates and synchronize

Manage Calculator module

Successfully updated

System is up to date

Generate Reports

Request for reports

View and download reports

Reports generated successfully

Logout

Terminate connection to database

Log user out of the system successfully

Logout successfully

###### Figure 4.7 Sequence Diagram for Administrator

##### 4.3.1.2.4 Class Diagrams

A class diagram shows an overview of the target system. It describes the objects and classes inside the system and the relationship between them. It also shows a wide variety of usages: from modeling the domain-specific data structure to detailed design of the target system. With the share model facilities, you can reuse your class model in the interaction diagram for modeling the detailed design of the dynamic behavior.

In the Smart Shopping application system we have four different classes that collaborate and work together in the system. The classes include: The Barcode Scanner, Calculator, Notepad and the system itself. These classes have different functions and methods within them that show how the different classes interact.

**Class Diagram**

System

Barcode Scanner

Shopping List

-Open()

-close()

-Amount

-Scanned list

-note 1

-note2

-note..n

+Scan()

+Rescan()

+pay()

+new entry()

+save()

+delete()

+open()

Calculator

-first number

-secondnumber

-result

+calculate()

###### Figure 4.8 Class Diagram

## 4.4 Smart Shopping Mobile Application System Implementation

### 4.4.1 Tools used

The tools to be used to develop the android-based application are very critical to consider. This is because of factors such as performance, storage, size of the application, security, portability etc. It is therefore important to be careful while choosing the appropriate tool.

#### 4.4.1.1 Software requirement

This application is designed and developed in the Android IDE called Android Studio. The user interface is therefore done using the extensible Markup Language (XML).

The logic of the application is done in the Java programming language, which is one of the languages supported by the IDE Android studio.

#### 4.4.1.2 External Libraries and APIs

The system imports and makes use of very useful software libraries such as the Zxing Library (Zebra Crossing), for enabling capability of making the phone camera behave as a barcode reader.

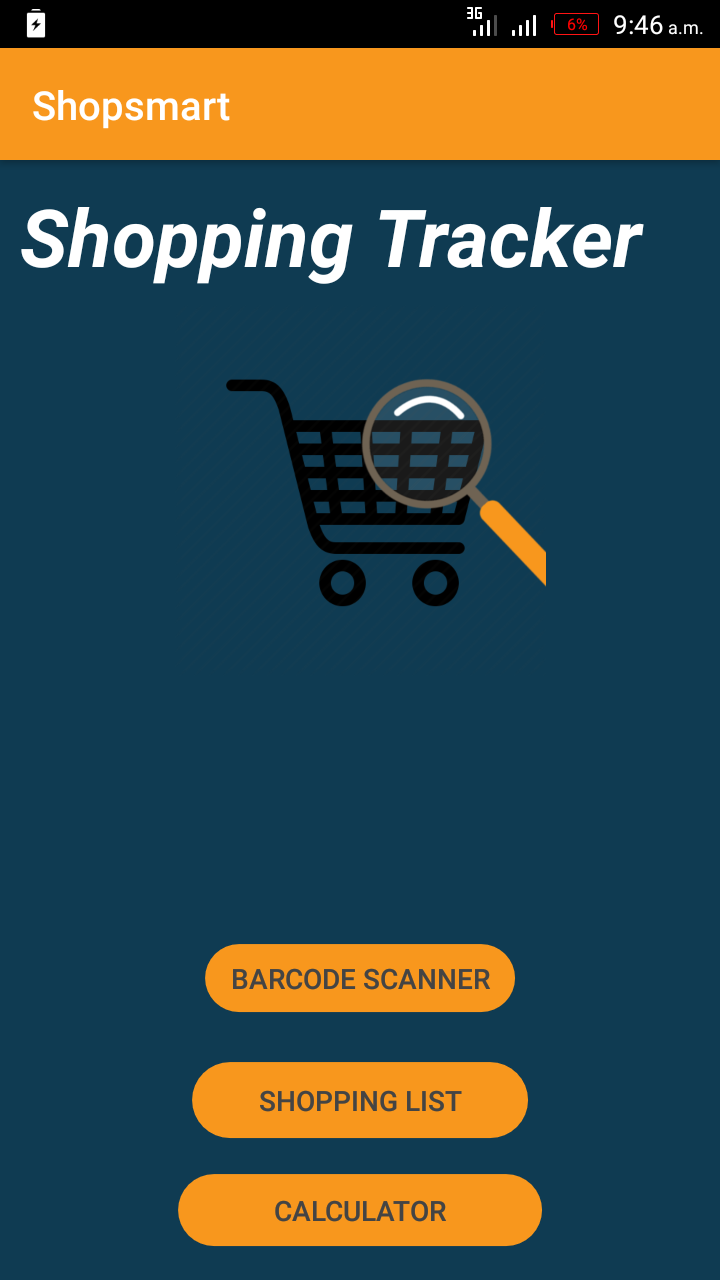
Other libraries and APIs include:

**4.4.1.3 Database management**

SQLite Database

* 1. **Graphical User Interface Home tab/activity**

The home tab is displayed when the app is opened. It contains three buttons from which the user can access the three modules of the system.

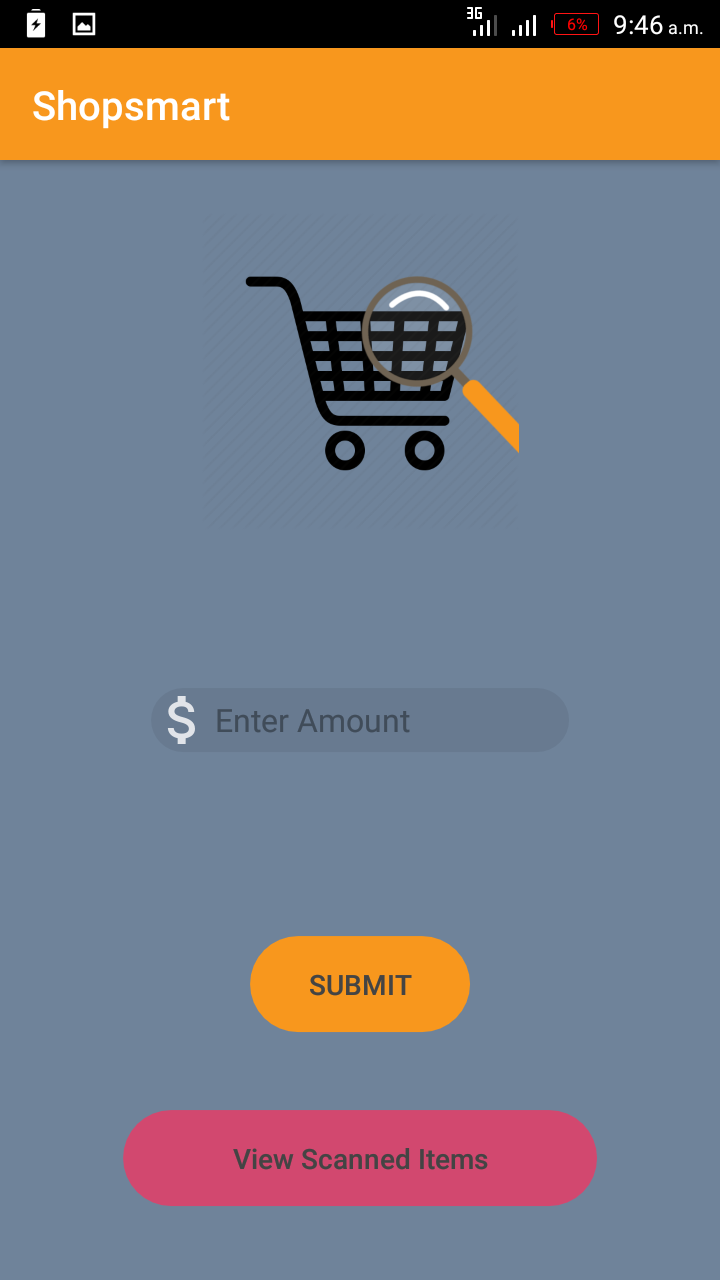


###### Figure 4.9 Home tab

* 1. **Enter Amount Tab**

After clicking the Barcode Scanner module, the user is prompted to enter an amount of money to be used in shopping. The system will manage it for them.

The user can view the shopping list from this page/tab.



###### Figure 4.10 Amount tab

* 1. **Barcode Scanner Tab**

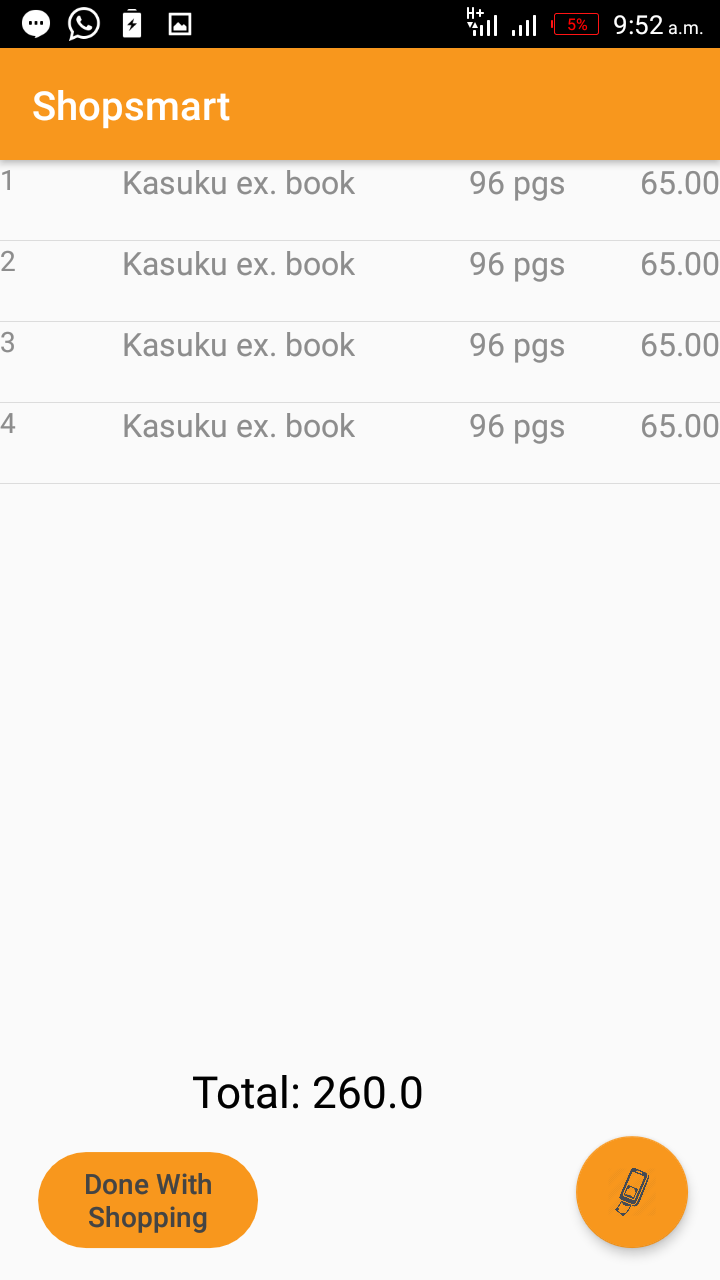
In this tab, the barcode scanner opens, and the user can be able to scan the barcode of items.



###### Figure 4.11 Barcode Scanner tab

* 1. **Scanned Lists tab**

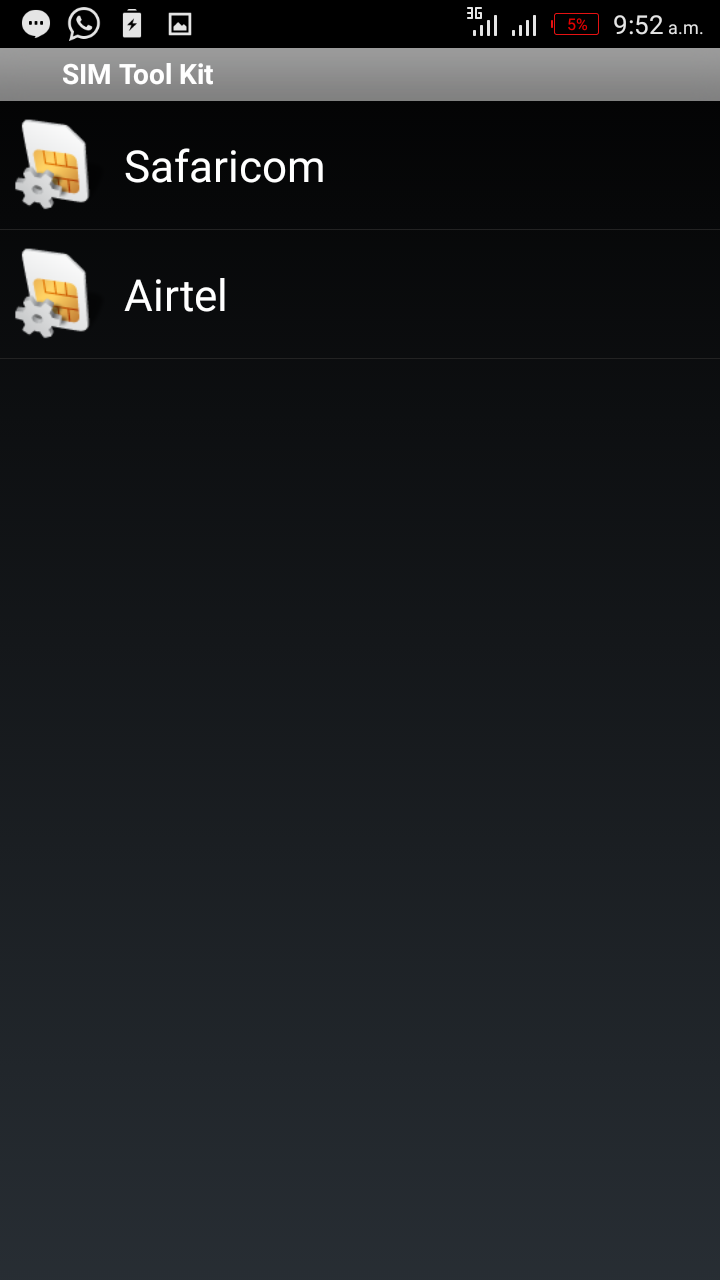
This tab displays the list of items that have been scanned by the user in a list view format. This enables the user to manage the scanned items by either adding more, or deleting an entry. The total amount of the items scanned is displayed for the user to always be in the know of their expenditures.



###### Figure 4.12 Scanned List tab

* 1. **Mobile payment tab**

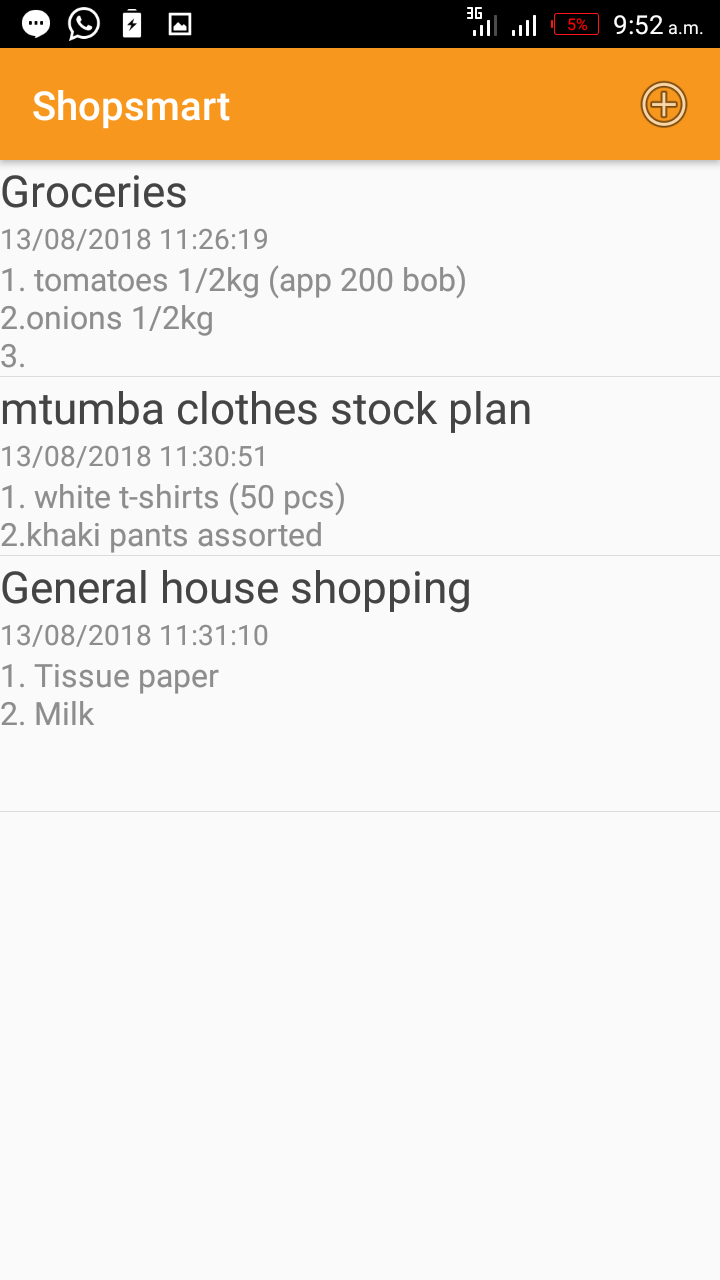
This tab is opened from the scanned lists tab, and it opens the sim tool kit for the user to pay for the total amount of items scanned using mobile money.



###### Figure 4.13 Mobile payment tab

* 1. **Shopping List Tab**

This module enables the user to create a new shopping list, edit an existing note and even delete an unwanted list. It helps a user to pre-plan a shopping activity by offering a platform to write a list before the actual shopping.



###### Figure 4.14 Shopping List tab

* 1. **Calculator Tab**

This module enables the user to have an easy access to a calculator when urgently needed and the user does not want to leave the app session. This will enable the user perform arithmetic calculations related to their shopping and hence be able to manage their money well. It acts as a backup in case the automatic barcode scanner does not work.



###### Figure 4.15 Calculator Tab

## 4.5 Testing

The testing was conducted in an iterative approach so the program design and implementation was continuously tested and evaluated thorough out the whole project. Input data was compared with the output given to check whether they were the required output. The following is a description of testing objectives, strategies that were employed during the testing of the system.

Smart Shopping mobile application testing objectives are;

1. To find errors or faults in software.
2. Test the system against user’s requirements.
3. To analyze the test results.

### 4.5.1 Testing Strategies

A test strategy is general approach to the testing process. This involved testing the system using different types of system tests. The process aimed at uncovering errors and measuring the system capability. The following system tests were performed on various areas during system development:

#### 4.5.1.1 Unit Testing

This was the first test in the development process and ensured that each unique part of the project performed accurately to the documented specifications and contained clearly defined inputs and expected results. The individual units in the system were tested separately so as to verify that they conform to the users’ requirements and operate according to the systems specification. Examples include: the enter amount tab denied user access to the barcode activity unless a valid amount was entered, when a barcode that does not exist in the database was scanned, an error message was displayed to the user. Unit testing treated each component as a stand-alone entity, which does not need other components during the testing process. Each module was tested to ascertain that it fully performed as expected.

**4.5.1.2 Integration Testing**

This involved testing integrated modules. Integration testing was done to ensure that modules could easily interact with each other and that the integration of the modules making up the system was working. The process involved a systematic technique for constructing the program structure while conducting tests to uncover errors associated with interfacing.

**4.5.1.3 Acceptance Testing**

This involves testing the system with the intent of confirming readiness of the product and customer acceptance.

#### 4.5.1.4 Alpha Testing

Alpha testing is done after coding. Alpha testing concentrated on verifications which included; ensuring valid entry and output for each record, system usability, amount entry validation, database and the interface consistency.

## 4.6 Chapter Summary

This chapter analyzed, interpreted and presented findings of the research. It discussed the functional and non-functional requirements of the application system. These requirements are then analyzed and transformed into design specifications and presented using UML diagrams. The chapter also discussed and outlined the software tools used in development of the system.

# CHAPTER FIVE

# SUMMARY, CONCLUSION AND RECOMMENDATIONS

## 5.1 Chapter Overview

This chapter provides a summary of the entire research findings, enables the researcher to make relevant conclusions and recommendations based on various objectives of the study.

The aim of the study was to design and develop a smart shopping application system for the android mobile smartphone. The application is intended for the use by anyone who occasionally goes to a supermarket to shop for items.

## 5.2 Summary of Findings

The first objective of the study was to adopt a suitable methodology and use it to develop a smart shopping mobile application system. In chapter 3, section 3.2 , the waterfall methodology is selected as the most convenient and appropriate methodology to design and develop the application system. This in summary is because the waterfall model identifies requirements long before programming begins.

The second objective of the study was to perform an analysis on requirements gathered for the application system. This has been achieved through the use of data collection tools to first collect the user requirements. The tools used in the study include Questionnaires, Interview questions and direct observations as specified in chapter 3 section 3.3. Analysis of the requirements is thoroughly done, especially the user requirements because the application is100% user oriented. See chapter3 section 3.10.

The third objective of the study was to provide a way of making shopping a very tireless, easy and enjoyable activity. This was to be achieved through ultimately designing and developing the application system. In the whole of chapter 4, the researcher and developer uses the requirements gathered in chapter 3, and analyzes it to come up with a design for the system. This designed is used to model and develop the actual application system using software tools. (See chapter 4 section 4.4.)

## 5.3 Conclusion

This research study looked at how people do their normal household shopping in supermarkets. It looked at the various challenges faced, the resources, tools and application systems directly and indirectly related to shopping activities.

It was established that over reliance on brain memory to recall all the items on a shopping list in most cases led to items being forgotten during the actual shopping. This meant that the shopper had to go back to the supermarket if this item is important and urgent.

It was also established in this study that doing mental calculations of prices of total items purchased and subtracting this from the cash at hand resulted in most individual cases, either overestimation or underestimation of their purchasing powers as most people could not accurately keep up with the math for long while all sorts of distractions are present in the supermarket.

Developing a smart shopping mobile application assistant will make people save on time, be accurate about their shopping, make shopping fun and interactive and relieve a person’s mind of the math that comes along with shopping.

## 5.4 Recommendation

**Networked application**

The application should in future be upgraded to fetch details of items from all the available supermarkets and compare them, and therefore give the user a price friendly option.

**Google Maps**

Google maps should be integrated in the system to give the user a view of the available supermarkets in the vicinity.

**Other Mobile OS platforms**

The application can be developed for other mobile-based platforms such as windows, blackberry and ios. This will enable a wider reach to more users and shoppers who are not on the android-based platform.

## 5.5 Suggestion for further research

1. **Integration with IOT technology**

The shopping assistant should be integrated with a smart home in the Internet of things technology, so as to always alert the user on items that have either replenished or expired.

1. **Real Time Data and Analytics technology in the app**

In a further research, the data and information being input, output and stored by this app can be tapped and utilized to give insight into customers and shoppers. Retailers will be able to know items in demand, best time in the year to stock certain items, which products to invest in advertising, potential purchasing power of a certain geographical region and also be able to determine if a new commodity can be successfully introduced to a market. This will in turn improve the shopping experience of people as shopping will become more personalized.

# REFERENCES

* Alba, J., Lynch, J., Weitz, B., Janiszewski, C., Lutz, R., Sawyer, A., & Wood, S. (1997). Interactive home shopping: consumer, retailer, and manufacturer incentives to participate in electronic marketplaces. *The Journal of Marketing*, 38-53.
* Android, O. S. (2011). Android. *Retrieved February*, *24*, 2011.
* Arthur, C. (2012). The history of smartphones: Timeline. *The Guardian*, *24*.
* Auger, C., Leduc, E., Labbé, D., Guay, C., Fillion, B., Bottari, C., & Swaine, B. (2014). Mobile Applications for Participation at the Shopping Mall: Content Analysis and Usability for Persons with Physical Disabilities and Communication or Cognitive Limitations. *International Journal of Environmental Research and Public Health*, *11*(12), 12777–12794. <http://doi.org/10.3390/ijerph111212777>
* Benington, H.D. (1956): Production of large computer programs. In Proceedings, ONR Symposium on Advanced Programming Methods for Digital Computers, June 1956, pp 15-27.
* Birrell, N. D. and Ould, M.A. (1988): A practical handbook to software
* Boulos, M. N. K., Wheeler, S., Tavares, C., & Jones, R. (2011). How smartphones are changing the face of mobile and participatory healthcare: an overview, with example from eCAALYX. *Biomedical engineering online*, *10*(1), 24.
* Davis, D. (2013). *A history of shopping*. Routledge.
* Fling, B. (2009). *Mobile design and development: Practical concepts and techniques for creating mobile sites and Web apps*. " O'Reilly Media, Inc.".
* Gefen, D., Karahanna, E., & Straub, D. W. (2003). Trust and TAM in online shopping: An integrated model. *MIS quarterly*, *27*(1), 51-90.
* <http://www.onlinemarketingdegree.net/resources/history-of-shopping-center/>
* <https://expertise.jetruby.com/brief-history-of-mobile-apps-286fbbf766a9>
* <https://manifesto.co.uk/history-mobile-application-development/>
* <https://retailminded.com/the-history-of-retail-shopping/>
* <https://steelkiwi.com/blog/grocery-chains-using-mobile-tech-modernize-shoppin/>
* <https://tech.co/mobile-app-history-evolution-2015-11>
* <https://www.droidviews.com/essay-the-history-of-android-os/>
* https://www.forbes.com/sites/quora/2017/12/19/why-many-online-shopping-sites-are-becoming-mobile-shopping-apps/
* <https://www.hongkiat.com/blog/mobile-shopping-apps/>
* <https://www.pcworld.com/article/199243/a_brief_history_of_smartphones.html>
* <https://www.pcworld.com/article/199243/a_brief_history_of_smartphones.html>
* Inukollu, V. N., Keshamoni, D. D., Kang, T., & Inukollu, M. (2014). Factors influencing quality of mobile apps: Role of mobile app development life cycle. *arXiv preprint arXiv:1410.4537*.
* [Jon Stobart](https://www.emeraldinsight.com/author/Stobart%2C+Jon), (2010) "A history of shopping: the missing link between retail and consumer revolutions", , Vol. 2 Issue: 3, pp.342-349,
* Lee, M. K., & Turban, E. (2001). A trust model for consumer internet shopping. *International Journal of electronic commerce*, *6*(1), 75-91.
* Lessard, J., & Kessler, G. (2010). Android Forensics: Simplifying Cell Phone Examinations.
* Maia, C., Nogueira, L. M., & Pinho, L. M. (2010, July). Evaluating android os for embedded real-time systems. In *6th international workshop on operating systems platforms for embedded real-time applications* (pp. 63-70).
* Micheletti, M. (2003). Shopping with and for Virtues. In *Political virtue and shopping* (pp. 149-168). Palgrave Macmillan, New York.
* Nguyen, Tuan C. "The History of Smartphones." ThoughtCo, Jun. 14, 2018, thoughtco.com/history-of-smartphones-4096585.
* Paek, J., Kim, J., & Govindan, R. (2010, June). Energy-efficient rate-adaptive GPS-based positioning for smartphones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services* (pp. 299-314). ACM.
* Raymond, Eric (2001): Cathedral and the Bazaar, 1st Edition; O'Reilly
* Royce, Winston W. (1970): Managing the development of large software development; Cambridge University Press. ISBN 978-0521347921. pp 3-12.
* Shields, R. (Ed.). (2003). *Lifestyle shopping: The subject of consumption*. Routledge.
* Stobart, J. (2008). *Spend, spend, spend! A history of shopping*. History Press.
* TOGAF 9:The Open Group; <http://www.opengroup.org/togaf/>.
* Tracy, K. W. (2012). Mobile application development experiences on Apple’s iOS and Android OS. *Ieee Potentials*, *31*(4), 30-34.

# APPENDICES

## Appendix 1: Letter to respondent/ site manager

I am a student from Moi University, main campus undertaking research on a smart shopping mobile application system. The aim of this research is to identify whether people actually practically need a shopping assistant in the form of an phone application.

Kindly assist in filling the questionnaire. Information provided will be used for purposes of the research only and will be treated with utmost confidentiality.

## Appendix 2: Questionnaire

**Section A: General Information about you**

1. Gender

Male

Female

1. Age

15 – 25 years 26-35 years 36-45 years 46 years and above

1. Shopping experience in years

0-2 years 3-5 years 6-10 years 10 years and above

**Section B: Open Ended Questions**

1. What are some of the general challenges you face while shopping?................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................
2. Do you find it easy mentally performing calculations as you shop? Explain………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………....
3. Do you write your shopping list on an actual piece of paper?.............................................................................................................................................................................................................................................................................................
4. What are some of the advantages associated with the new system?...................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................
5. What are some of the challenges that might be experienced in the event of adopting this shopping application?.................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................
6. Kindly suggest ways of improving this shopping application system to suit your needs……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………....
7. General comments about the system?......................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

## Appendix 3: System requirements

**Hardware requirement**

The selection of a suitable Hardware and software is very important. This is to enable the system to run smoothly and work efficiently.

**Hardware**

A smartphone or tablet.

**Software Requirements**

Android OS 6.0 and above.

## Appendix 4: Project Codes

**Sample code for Database Tables and query functions**

public class DatabaseHelper extends SQLiteOpenHelper {

private static final int *DATABASE\_VERSION* = 26;

private static final String *DATABASE\_NAME* = "contactsManager";

//private SQLiteDatabase db;

String SQL\_CREATE\_SCANNED\_ITEM\_TABLE = "CREATE TABLE " + ColumnsContract.ScannedItemEntry.*TABLE\_SCANNED\_ITEMS* + " ("

+ ColumnsContract.ScannedItemEntry.*COLUMN\_ITEM\_ID* + " INTEGER PRIMARY KEY AUTOINCREMENT, "+ ColumnsContract.ScannedItemEntry.*COLUMN\_ITEM\_NAME* + " TEXT, "+ ColumnsContract.ScannedItemEntry.*COLUMN\_ITEM\_QTY* + " TEXT, " + ColumnsContract.ScannedItemEntry.*COLUMN\_SCANNED\_ITEMS\_PRICE* + " TEXT NOT NULL " + ")";

String SQL\_DROP\_SCANNED\_ITEM\_TABLE = "DROP TABLE IF EXISTS "+ColumnsContract.ScannedItemEntry.*TABLE\_SCANNED\_ITEMS*;

String SQL\_CREATE\_BARCODES\_TABLE = "CREATE TABLE " + ColumnsContract.BarcodesReference.*TABLE\_BARCODE* + " ("

+ ColumnsContract.BarcodesReference.*COLUMN\_BARCODE* + " TEXT, " + ColumnsContract.BarcodesReference.*COLUMN\_ITEM\_NAME* + " TEXT, "

+ ColumnsContract.BarcodesReference.*COLUMN\_ITEM\_PRICE* + " TEXT, " +

ColumnsContract.BarcodesReference.*COLUMN\_QUANTITY* + " TEXT NOT NULL " + ")";

String SQL\_DROP\_BARCODES\_TABLE = "DROP TABLE IF EXISTS "+ColumnsContract.BarcodesReference.*TABLE\_BARCODE*;

public DatabaseHelper (Context context){

super(context, *DATABASE\_NAME*, null, *DATABASE\_VERSION*);

}

@Override

public void onCreate(SQLiteDatabase db) {

db.execSQL(SQL\_CREATE\_SCANNED\_ITEM\_TABLE);

db.execSQL(SQL\_CREATE\_BARCODES\_TABLE);

Log.*e*("onCreate: ", "Payment CREATED");

Log.*e*("onCreate: ", "Barcodes CREATED");

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

db.execSQL(SQL\_DROP\_SCANNED\_ITEM\_TABLE);

db.execSQL(SQL\_DROP\_BARCODES\_TABLE);

onCreate(db);

}

public long insertScannedItem(BarcodeQueryResponse response){

SQLiteDatabase db = this.getWritableDatabase();

ContentValues values = new ContentValues();

values.put(ColumnsContract.ScannedItemEntry.*COLUMN\_ITEM\_QTY*, response.getItemQuantity());

values.put(ColumnsContract.ScannedItemEntry.*COLUMN\_ITEM\_NAME*, response.getItemName());

values.put(ColumnsContract.ScannedItemEntry.*COLUMN\_SCANNED\_ITEMS\_PRICE*, response.getItemPrice());

long result = db.insert(ColumnsContract.ScannedItemEntry.*TABLE\_SCANNED\_ITEMS*,null,values);

// Log.e("saved\_item\_name", name);

Log.*e*("savedPayments", ""+result);

db.close();

return result;

}

public void insertBarcodesItem(JSONArray barcodeArray) throws JSONException{

SQLiteDatabase db = this.getWritableDatabase();

for (int i = 0; i<barcodeArray.length();i++) {

ContentValues values = new ContentValues();

values.put(ColumnsContract.BarcodesReference.*COLUMN\_BARCODE*, barcodeArray.getJSONObject(i).getString("barcode"));

values.put(ColumnsContract.BarcodesReference.*COLUMN\_ITEM\_NAME*, barcodeArray.getJSONObject(i).getString("itemName"));

values.put(ColumnsContract.BarcodesReference.*COLUMN\_ITEM\_PRICE*, barcodeArray.getJSONObject(i).getString("itemPrice"));

values.put(ColumnsContract.BarcodesReference.*COLUMN\_QUANTITY*, barcodeArray.getJSONObject(i).getString("itemQuantity"));

long result = db.insert(ColumnsContract.BarcodesReference.*TABLE\_BARCODE*, null, values);

Log.*e*("Saved\_\_Items", "" + result);

}

db.close();

}

public ArrayList<BarcodeQueryResponse> getAllAmounts(){

SQLiteDatabase db = this.getWritableDatabase();

ArrayList<BarcodeQueryResponse> data = new ArrayList<>();

String query = "SELECT \* FROM "+ ColumnsContract.ScannedItemEntry.*TABLE\_SCANNED\_ITEMS*;

Cursor cursor = db.rawQuery(query,null);

while (cursor.moveToNext()){

BarcodeQueryResponse barcodeQueryResponse = new BarcodeQueryResponse();

barcodeQueryResponse.setBarcodeName(cursor.getString(cursor.getColumnIndex(ColumnsContract.ScannedItemEntry.*COLUMN\_ITEM\_ID*)));

barcodeQueryResponse.setItemName(cursor.getString(cursor.getColumnIndex(ColumnsContract.ScannedItemEntry.*COLUMN\_ITEM\_NAME*)));

barcodeQueryResponse.setItemPrice(cursor.getString(cursor.getColumnIndex(ColumnsContract.ScannedItemEntry.*COLUMN\_SCANNED\_ITEMS\_PRICE*)));

barcodeQueryResponse.setItemQuantity(cursor.getString(cursor.getColumnIndex(ColumnsContract.ScannedItemEntry.*COLUMN\_ITEM\_QTY*)));

data.add(barcodeQueryResponse);

}

db.close();

return data;

}

public BarcodeQueryResponse getBarcodeDetail(String barcodes){

SQLiteDatabase db = this.getWritableDatabase();

BarcodeQueryResponse result = new BarcodeQueryResponse();

// String query = "SELECT \* FROM "+ ColumnsContract.BarcodesReference.TABLE\_BARCODE +" WHERE "+ColumnsContract.BarcodesReference.COLUMN\_BARCODE + "='"+"1111111"+"'";

String query = "SELECT \* FROM "+ ColumnsContract.BarcodesReference.*TABLE\_BARCODE* +" WHERE "+ColumnsContract.BarcodesReference.*COLUMN\_BARCODE* + " = '"+barcodes+"'";

Cursor cursor = db.rawQuery(query,null);

while (cursor.moveToNext()){

result.setBarcodeName(cursor.getString(cursor.getColumnIndex(ColumnsContract.BarcodesReference.*COLUMN\_BARCODE*)));

result.setItemName(cursor.getString(cursor.getColumnIndex(ColumnsContract.BarcodesReference.*COLUMN\_ITEM\_NAME*)));

result.setItemPrice(cursor.getString(cursor.getColumnIndex(ColumnsContract.BarcodesReference.*COLUMN\_ITEM\_PRICE*)));

result.setItemQuantity(cursor.getString(cursor.getColumnIndex(ColumnsContract.BarcodesReference.*COLUMN\_QUANTITY*)));

}

db.close();

return result;

}

public void deleteAScannedEntry(int id){

SQLiteDatabase db = this.getWritableDatabase();

String query = "DELETE FROM " + ColumnsContract.ScannedItemEntry.*TABLE\_SCANNED\_ITEMS* + " WHERE " +

ColumnsContract.ScannedItemEntry.*COLUMN\_ITEM\_ID* + " = '" + id + "'";

db.execSQL(query);

}

public void deleteAll()

{

SQLiteDatabase db = this.getWritableDatabase();

db.delete(ColumnsContract.ScannedItemEntry.*TABLE\_SCANNED\_ITEMS*,null,null);

db.execSQL("DELETE FROM "+ ColumnsContract.ScannedItemEntry.*TABLE\_SCANNED\_ITEMS*);

db.close();

}

}

**Sample code for Barcode scanner implementation**

public class BarcodeActivity extends AppCompatActivity implements ZXingScannerView.ResultHandler,Serializable {

private static final int *REQUEST\_CAMERA* = 1;

private ZXingScannerView scannerView;

private static int *camId* = Camera.CameraInfo.*CAMERA\_FACING\_BACK*;

DatabaseHelper myDb;

public static Methods *myMethods*;

// static int totalAmount;

public static Double *totalAmount*;

static Double *x*;

//static int x;

static Double *totalBill* = 0.0;

public static AlertDialog.Builder *builder*;

TextView textView;

static BarcodeQueryResponse *itemDetais*;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

*builder* = new AlertDialog.Builder(this);

*itemDetais* = new BarcodeQueryResponse();

myDb = new DatabaseHelper(this);

*myMethods* = new Methods(this);

*builder*.setTitle("Scan Result" + "\nAmount Remaining : " + Double.*parseDouble*(*myMethods*.getAmount()));

textView = (TextView) findViewById(R.id.*tvTotalDisplay*);

scannerView = new ZXingScannerView(this);

setContentView(scannerView);

int currentApiVersion = Build.VERSION.*SDK\_INT*;

if(currentApiVersion >= Build.VERSION\_CODES.*M*)

{

if(checkPermission())

{

Toast.*makeText*(getApplicationContext(), "Scan an Item's Barcode!", Toast.*LENGTH\_LONG*).show();

}

else

{

requestPermission();

}

}

}

private boolean checkPermission()

{

return (ContextCompat.*checkSelfPermission*(getApplicationContext(), *CAMERA*) == PackageManager.*PERMISSION\_GRANTED*);

}

private void requestPermission()

{

ActivityCompat.*requestPermissions*(this, new String[]{*CAMERA*}, *REQUEST\_CAMERA*);

}

@Override

public void onResume() {

super.onResume();

int currentapiVersion = android.os.Build.VERSION.*SDK\_INT*;

if (currentapiVersion >= android.os.Build.VERSION\_CODES.*M*) {

if (checkPermission()) {

if(scannerView == null) {

scannerView = new ZXingScannerView(this);

setContentView(scannerView);

}

scannerView.setResultHandler(this);

scannerView.startCamera();

} else {

requestPermission();

}

}

}

@Override

public void onDestroy() {

super.onDestroy();

scannerView.stopCamera();

}

public void onRequestPermissionsResult(int requestCode, String permissions[], int[] grantResults) {

switch (requestCode) {

case *REQUEST\_CAMERA*:

if (grantResults.length > 0) {

boolean cameraAccepted = grantResults[0] == PackageManager.*PERMISSION\_GRANTED*;

if (cameraAccepted){

Toast.*makeText*(getApplicationContext(), "Permission Granted, Now you can access camera", Toast.*LENGTH\_LONG*).show();

}else {

Toast.*makeText*(getApplicationContext(), "Permission Denied, You cannot access and camera", Toast.*LENGTH\_LONG*).show();

if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*M*) {

if (shouldShowRequestPermissionRationale(*CAMERA*)) {

showMessageOKCancel("You need to allow access to both the permissions",

new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*M*) {

requestPermissions(new String[]{*CAMERA*},

*REQUEST\_CAMERA*);

}

}

});

return;

}

}

}

}

break;

}

}

private void showMessageOKCancel(String message, DialogInterface.OnClickListener okListener) {

new android.support.v7.app.AlertDialog.Builder(BarcodeActivity.this)

.setMessage(message)

.setPositiveButton("OK", okListener)

.setNegativeButton("Cancel", null)

.create()

.show();

}

@Override

public void handleResult(final Result result) {

final String myResult = result.getText();

Log.*d*("QRCodeScanner", result.getText());

Log.*d*("QRCodeScanner", result.getBarcodeFormat().toString());