

A Blockchain Based Mobile Application For Land Ownership Record And Product Anti - Counterfeiting

A PROJECT REPORT

Submitted by

SARWER HUSSAIN A [REGISTER NO:211418104233]

in partial fulfillment for the award of the

degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



PANIMALAR ENGINEERING COLLEGE

**(An Autonomous Institution, Affiliated to Anna University,
Chennai)**

MAY 2022

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **A Blockchain Based Mobile Application For Land Ownership Record And Product Anti - Counterfeiting** the bonafide work of **Sarwer Hussain A (211418104233)** , who carried out the project work under my supervision.

SIGNATURE

Dr.S.MURUGAVALLI,M.E.,Ph.D.,
HEAD OF DEPARTMENT

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLE,
CHENNAI – 6000123.

SIGNATURE

Dr. N. PUGHAZENDI, M.E.,Ph.D.,
PROFESSOR

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLE,
CHENNAI – 6000123.

Certified that the above mentioned students were examined in End Semester project viva-voice held on _____.

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION BY THE STUDENT

I **Sarwer Hussain A [211418104233]**, hereby declare that this project report titled **A Blockchain Based Mobile Application For Land Ownership Record And Product Anti - Counterfeiting** , under the guidance of **Dr. N. Pughazendi, M.E.,Ph.D** is the original work done by me and we have not plagiarized or submitted to any other degree in any university by me.

ACKNOWLEDGEMENT

I would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

I express our sincere thanks to our Directors **Tmt.C.VIJAYA RAJESWARI, Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

I also express our gratitude to our Principal **Dr.K.MANI, M.E., Ph.D.** who facilitated us in completing the project.

I thank the Head of the CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

I would like to thank my **Dr. N. PUGHAZENDI M.E.,Ph.D.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

I would like to thank my Parents, **AFSAR HUSSAIN** and **LUBNA BANU** for their extended support throughout the project.

ABSTRACT

Nowadays with the advancement of technology people come up with new ways of duplicating a product and selling it for the same price as the original. Same is the case with land fraud as there are cases of forced land acquiring through fake documents. This project aims to solve this problem by providing a mobile application with Blockchain technology as the base architecture ensures that the contents of its data are tamper-proof. The mobile application displays a list of sources distributed by the product/country, so you can be sure that your end distributor is selling a genuine product. Blockchain guarantees transparency and reliability because the technology is decentralized over a shared network. That is, you cannot change the data. You can take ownership of land / real estate in the same application. In this case, large parties such as governments can act as a source to verify reliability and place data on the blockchain through applications. Therefore, a digital signature is assigned to the owner to ensure the security of the owner's ownership. As blockchain is a decentralized and distributed network there is no single point of failure, which makes it much harder to corrupt. Hacking into one part of the system cannot affect other parts. Hence it maintains the asset ownership of the user. The proposed system involves the use of a mobile application for the end user along with Blockchain technology in the backend as the base architecture ensures that the contents of its data are tamper proof.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
	ABSTRACT	v
	LIST OF FIGURES	vii
	LIST OF ABBREVIATIONS	viii
1.	INTRODUCTION	1
	1.1 INTRODUCTION	1
	1.2 AIM OF THE PROJECT	1
	1.3 PROJECT DOMAIN	2
2.	LITERATURE SURVEY	11
3.	SYSTEM ANALYSIS	17
	3.1 EXISTING SYSTEM	17
	3.2 PROPOSED SYSTEM	18
	3.3 REQUIREMENT SPECIFICATION	18
	3.4 PLATFORM SPECIFICATION	19
4.	SYSTEM DESIGN	24
	4.1 ER DIAGRAM	23
	4.2 DATA FLOW DIAGRAM	24
	4.3 USE CASE	25
5.	MODULE DESCRIPTION	29
	5.1 CREATION OF SMART CONTRACT	30
	5.2 INTEGRATION WITH APPLICATION	30
	5.3 PREDICTION OF OUTPUT	31
6.	TESTING	32
	6.1 TEST CASE	
7	CONCLUSION	33
8	SYSTEM IMPLICATION	35
9	APPENDICES	55
10	REFERENCES	57

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
4.1	ER Diagram	24
4.2	Data Flow Diagram 0	25
4.3.1	Use Case	26
4.3.2	Activity Diagram	27
4.3.3	Sequence Diagram	28
4.3.4	Class Diagram	29
5.2.2	Ganache	31
5.4	Application	32
9.1	Admin App	55
9.2	User App	56

LIST OF ABBREVIATION

DLT	Distributed Ledger Technology
POW	Proof Of Work
POS	Proof Of Stake
POS	Point Of Sale
dApps	Decentralized Apps
SCM	Supply Chain Management

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

The authenticity and ownership of the product has always been at risk of being tampered as the records can easily be duplicated and modified or a fake product can be replaced with it. Hence to solve this problem Blockchain technology is used as the data which are uploaded in the chain cannot be tampered or changed. As blockchain is a decentralized and distributed network there is no single point of failure, which makes it much harder to corrupt. Hacking into one part of the system cannot affect other parts. Hence it maintains the asset ownership of the user.

1.2 AIM OF THE PROJECT

This project aims to solve this problem by providing a mobile application with Blockchain technology as the base architecture ensures that the contents of its data are tamper-proof. The proposed system involves the use of a mobile application for the end user along with Blockchain technology in the backend as the base architecture ensures that the contents of its data are tamper proof. The mobile application will allow the user to scan the qr code on the product. On scanning, the app will display a list of the verified distributors through which the product has been. Since every distributor has to scan and transfer the ownership before selling to the next distributor. Hence the app tracks the authenticity of a product. Finally the end user can scan and verify the manufacturer and the last owner of the product which should be the seller. Hence, authenticity of product is ensured.

1.3 PROJECT DOMAIN

1.3.1 Blockchain

A blockchain is a distributed database shared by computer network nodes. A blockchain acts as a database, storing data in an electronic manner. Blockchains are best recognised for preserving a secure and decentralized record of transactions in cryptocurrency systems like Bitcoin. A blockchain's novelty is that it ensures the authenticity and security of a data record while also generating trust without the need for a third party.

1.3.2 Types Of Blockchain

Public Blockchain

The public blockchain is the first form of blockchain technology. This is where Bitcoin and other cryptocurrencies first appeared, and where they helped to promote distributed ledger technology (DLT). It eliminates the drawbacks of centralization, such as a lack of security and transparency. DLT distributes data throughout a peer-to-peer network rather than storing it in a single location. Its decentralised nature necessitates some kind of data authentication. This approach is a consensus process in which blockchain users agree on the present state of the ledger. Two typical consensus approaches are proof of work (PoW) and proof of stake (PoS).

Anyone with internet connectivity may join on to a blockchain platform and become an authorised node, making public blockchain non-restrictive and permissionless. This user has access to current and historical data, as well as the ability to perform mining operations, which are sophisticated calculations needed to validate transactions and add them to the ledger. On the network, no legitimate record or transaction may be modified, and because the source code is typically open source, anybody can check the transactions, uncover errors, and offer fixes.

Private Blockchain

A private blockchain is a blockchain network that operates in a restricted context, such as a closed network, or is controlled by a single company. While it functions similarly to a public blockchain network in terms of peer-to-peer connectivity and decentralisation, this blockchain is substantially smaller. Private blockchains, rather than allowing everyone to join and donate computer power, are often run on a limited network within a corporation or organisation. Permissioned blockchains and business blockchains are other names for them. Permission levels, security, authorizations, and accessibility are all determined by the governing organisation.

An enterprise putting up a private blockchain network, for example, can choose which nodes have access to view, contribute, or update data. It can also prohibit access to particular information by third parties. Private blockchains may be incredibly fast and execute transactions significantly faster than public blockchains due to their smaller size. The source code from private blockchains is often proprietary and closed. Users can't independently audit or confirm it, which can lead to less security. There is no anonymity on a private blockchain, either.

Hybrid Blockchain

Organizations who desire the best of both worlds will sometimes employ hybrid blockchain, a kind of blockchain that includes characteristics of both private and public blockchain. It allows businesses to build up a private, permission-based system alongside a public, permissionless system, letting them to regulate who has access to certain data stored on the blockchain and what data is made publicly available.

In a hybrid blockchain, transactions and information are often not made public but may be validated when necessary, such as by granting access through a smart contract. Inside the network, confidential information is kept secure yet still verifiable. Even if a private entity owns the hybrid blockchain, it is unable to modify transactions.

Real estate is one of the many applications for hybrid blockchain. Companies can utilize a hybrid blockchain to run systems securely while exposing some information to the public, such as listings. Retail may use hybrid blockchain to improve its procedures, and highly regulated industries like financial services can profit from it as well. Medical records may be maintained on a hybrid blockchain. Users may access their information using a smart contract, which can't be read by random third parties. Governments might potentially utilize it to keep citizen data securely while sharing it with other entities.

Consortium Blockchain

Consortium blockchains, also known as federated blockchains, are similar to hybrid blockchains in that they include private and public blockchain capabilities. However, it differs in that it involves various organisational members working together on a decentralised network. A consortium blockchain is essentially a private blockchain with limited access to a specific group, minimising the hazards associated with a private blockchain with only one company managing the network.

The consensus methods in a consortium blockchain are controlled by predetermined nodes. It has a validator node that handles transaction initiation, receipt, and validation. Transactions can be received or initiated by member nodes.

This sort of blockchain is used in banking and payments. Different banks can join forces to establish a consortium, which will decide which nodes will validate transactions. Food tracking groups, as well as research organisations, can construct a comparable model. It's perfect for supply chains, especially in the food and pharmaceutical industries. There are other consensus algorithms to consider, in addition to the four primary types of blockchain. Aside from PoW and PoS, anyone setting up a network should think about the other varieties, which are accessible on platforms like Wave and Burstcoin.

1.3.3 Consensus Mechanism

By consensus, we mean that everyone has agreed on something. Consider a bunch of individuals attending a movie. A consensus is reached if there is no dispute about a proposed film option. In the worst-case scenario, the gang will break up.

In the case of blockchain, the process is codified, and obtaining consensus implies that at least 51% of the network's nodes agree on the network's next global state.

Many people confuse consensus protocols and consensus algorithms. Protocols and algorithms, on the other hand, are distinct. A protocol is a collection of rules described in a standard that regulates the operation and interaction of a system and its many components. Algorithms are like step-by-step instructions for solving a problem or calculating a result.

A cryptoeconomic system's consensus method also aids in the prevention of certain types of economic assaults. By controlling 51 percent of the network, an attacker can theoretically jeopardise consensus. Consensus measures are in

place to prevent this "51 percent assault." Different approaches have been developed to address this security issue in various ways.

1.3.4 Types of Consensus Mechanism

Proof-of-work

Crypto currencies like Ethereum, Bitcoi uses a proof-of-work consensus protocol. Miners compete to build new blocks full of completed transactions, which is known as proof-of-work. The winner wins some freshly minted ETH and shares the new block with the rest of the network. The computer that solves a math challenge the fastest wins the race, which creates the cryptographic connection between the current block and the previous block. The effort in "proof-of-work" is solving this puzzle.

Blocks are formed from Ethereum transactions.

In every block

- 3,324,092,183,262,715 is an example of a block difficulty.
- MixHash 0x44bca881b07a6a09f8 3b130798072441705d 9a665c5ac8bdf2f39a3cdf3bee29
- Nonce – for example, 0xd3ee432b4fb3d26b

Proof-of-work is intimately tied to this block data.

Proof-of-Stake

Validators who have staked ETH to participate in the system do proof-of-stake. To produce new blocks, share them with the network, and collect rewards, a validator is picked at random. You only need to stake your ETH in the network instead of doing intensive computational effort. This is what encourages good network behavior. The fact that you'd need 51 percent of the total staked ETH to

scam a proof-of-stake system keeps it safe. And that your stake has been reduced due to malicious behavior.

Proof-of-stake comes with a number of improvements to the proof-of-work system:

- Better energy efficiency – there is no need to use lots of energy on proof-of-work computations
- Lower barriers to entry, reduced hardware requirements – there is no need for elite hardware to stand a chance of creating new blocks
- Reduced centralization risk – proof-of-stake should lead to more nodes securing the network
- Because of the low energy requirement less ETH issuance is required to incentivize participation
- Economic penalties for misbehaviour make 51% style attacks exponentially more costly for an attacker compared to proof-of-work
- The community can resort to social recovery of an honest chain if a 51% attack were to overcome the crypto-economic defenses.

1.3.5 Smart Contracts

A smart contract is a self-executing contract in which the conditions of the buyer-seller agreement are put directly into lines of code. The code and agreements it contains are disseminated throughout a decentralized blockchain network. Transactions are trackable and irreversible, and the programming regulates their execution.

Smart contracts eliminate the need for a central authority, legal system, or external enforcement mechanism to carry out trustworthy transactions and agreements between distant, anonymous participants.

Nick Szabo, an American computer scientist who devised a virtual currency called "Bit Gold" in 1998, roughly ten years before bitcoin, introduced smart contracts in 1994. In reality, Szabo has been accused of being the genuine Satoshi Nakamoto, the anonymous bitcoin creator, which he has denied. Smart contracts, according to Szabo, are automated transaction protocols that carry out the provisions of a contract. He sought to take electronic transaction methods like POS (point of sale) and bring them into the digital arena.

- Smart contracts are self-executing contracts in which the contents of the buyer-seller agreement are inscribed directly into lines of code.
- According to Nick Szabo, an American computer scientist who devised a virtual currency called "Bit Gold" in 1998, Smart contracts are computerized transaction protocols that execute contract conditions.
- Using it makes the transactions traceable, transparent, and irreversible.

Because the smart contract is incorporated on the blockchain, it cannot be lost. Smart contracts are only as accurate as the programmer who coded them for execution. Smart contracts employ software code to automate functions, lowering the amount of time required to navigate through all of the human contact procedures. Because everything is programmed, the time it takes to complete all of the tasks is the time it takes for the smart contract's code to run. The shared ledger is maintained by each node in the blockchain, giving the finest backup facility. There is no third component. You create the contract and share it with the other parties. There are no intermediaries, which reduces bullying and gives the negotiating parties complete control. Furthermore, all nodes on the network maintain and execute the smart contract, eliminating any control power from any one party. Security: Cryptography can ensure that the assets are secure. Even if the encryption is broken, the hacker will have to change all the blocks following the one that was changed. Please keep in mind

that this is a very tough and time-consuming process that is almost impossible for a small or medium-sized business to do. Savings: Smart contracts save money because they eliminate the need for intermediaries. Furthermore, the cost of documentation is little to none.

1.3.6 Decentralized Applications

Decentralized apps (dApps) are digital programmes or applications that operate on a blockchain or peer-to-peer (P2P) network of computers rather than a single computer. DApps (also known as "dapps") exist outside of the control of a single authority. DApps, which are frequently constructed on the Ethereum platform, may be used for a wide range of functions, including gaming, banking, and social networking. A normal web app, such as Uber or Twitter, operates on a computer system owned and maintained by a company, allowing them complete control over the app and its functionality. On one side, there may be several users, but the backend is managed by a single company.

DApps can be run on either a peer-to-peer or a blockchain network. BitTorrent, Tor, and Popcorn Time, for example, are software that operate on computers that are part of a P2P network, in which numerous users are consuming, feeding, or seeding content, or doing both roles at the same time.

Difference Between a Centralized and Decentralized App

A single corporation owns a centralised app. A centralised app's application software is stored on one or more company-controlled servers. You'll engage with the app like a user by downloading a copy of it and then submitting and receiving data from the company's server.

A decentralized app (also known as a dApp or dapp) is a computer programme

that runs on a blockchain or peer-to-peer network. It allows users to conduct transactions with one another without depending on a central authority. The user of a dApp will pay a cryptocurrency fee to the developer in order to obtain and utilise the program's source code. A smart contract is the source code that lets users to interact with it.

Twitter, Facebook, Instagram, and Netflix are all instances of centralised applications. Customers may access their accounts online using unified applications provided by banks and other financial organisations.

Peepeth, a Twitter-like social network, is an example of a decentralised programme. Cryptokitties is a decentralised application that lets users purchase and sell virtual kittens. MakerDAO is a decentralised credit facility that allows customers to create a collateralized debt position using the stablecoin Dai (CDP).

CHAPTER 2

LITERATURE REVIEWS

TITLE 1: Designing blockchain systems to prevent counterfeiting in wine supply chains: a multiple-case study

AUTHORS: Pamela Danese, Riccardo Mocellin, Pietro Romano

JOURNAL: International Journal of Operations & Production Management

YEAR: 17 December 2021

DESCRIPTION: It identifies three broad categories of approaches that companies use to mitigate product counterfeiting: product/packaging-related, customer information/education-related and process-related measures.

Like track-and-trace systems, BC can be considered a measure to prevent both upstream and downstream counterfeiting (Schmidt and Wagner, 2019). Unlike the other measures and in common with traditional track-and-trace systems, it allows customers to autonomously verify the authenticity of each product without the need to involve specific equipment/competencies or perform destructive chemical/physical/organoleptic analyses.

TITLE 2: Blockchain Technology Implementation in Logistics

AUTHORS: Edvard Tijan; Saša Aksentijević; Katarina Ivanić; Mladen Jardas

JOURNAL: International Journal of Operations & Production Management

YEAR: 23 February 2019

DESCRIPTION: The supply chain is tied to the complex processes of creation and distribution of goods. Depending on the product, the supply chain can include many phases, multiple geographic locations, several accounts and payments, several individuals, entities, and means of transport. Therefore, procurement of supplies can be extended over several months. Because of the complexity and the lack of transparency of traditional supply chains, it is of great interest for the stakeholders involved in the logistics process to introduce and develop blockchain technology to enhance the logistics processes in the supply chain, making them more sustainable

TITLE 3: Blockchain-empowered sustainable manufacturing and product lifecycle management in industry 4.0: A survey

AUTHORS: JiewuLeng; GuoleiRuan ;PingyuJiang; KailinXu; QiangLiu; XueliangZhou; ChaoLiue

JOURNAL: International Journal of Operations & Production Management

YEAR: 16 July 2020

DESCRIPTION: The transparency characteristics enabled by blockchain shows promising for enhancing the sustainability of manufacturing networks. Sustainability is a pressing need, as well as an engineering challenge, in the modern world. Developing smart technologies is a critical way to ensure that future manufacturing systems are sustainable. Blockchain is a next-generation development of information technology for realizing sustainability in businesses and industries. Much research on blockchain-empowered sustainable manufacturing in Industry 4.0 has been conducted from technical, commercial, organizational, and operational perspectives.

TITLE 4: Blockchain Technology in Healthcare: A Comprehensive Review and Directions for Future Research

AUTHORS: Seyednima Khezr ; Md Moniruzzaman ; Abdulsalam Yassine; Rachid Benlamri

JOURNAL: International Journal of Operations & Production Management

YEAR: 14 January 2019

DESCRIPTION: Blockchain technology is redefining data modeling and governance deployed in many healthcare applications. This is mainly due to its adaptability and abilities to segment, secure and share medical data and services in an unprecedented way. Blockchain technology is at the centre of many current developments in the healthcare industry. Emerging blockchain-based healthcare technologies are conceptually organized into four layers, including data sources, blockchain technology, healthcare applications, and stakeholders. Emerging blockchain-based healthcare technologies are conceptually organized into four layers, including data sources, blockchain technology, healthcare applications, and stakeholders

TITLE 5: Anti-Counterfeit Product System Using Blockchain Technology

AUTHORS: Anti-Counterfeit Product System Using Blockchain Technology

JOURNAL: International Journal of Operations & Production Management

YEAR: 5 October 2020

DESCRIPTION: Whenever a user requests a transaction on blockchain, the requested transaction is passed to the P2P network where it is broadcasted to the

nodes on the network There will be a user section where each user who is registered with the network has the ability to find the details of all the products that are owned by them. Also, they will have the authority to transfer the ownership of the product if they want to sell the product to someone else. There will be a feature given To the user to scan the QR code or enter the product id to fetch the details of that particular product which they want to buy in Future or want to check the authenticity and other information of the product. This helps user develop a sense of trust and reliability.

TITLE 6: Blockchain Technology for Supply Chain Management

AUTHORS: Taner Dursun

JOURNAL: Industrial Engineering in the Internet-of-Things World

YEAR: January 2022

DESCRIPTION: Blockchain technology offers important opportunities for the supply chain management. This paper aims to overview the employment of blockchain technology in the field of the supply chain. Although the technology has been widely associated with cryptocurrencies, non-financial applications such as supply chain, power, and food industry are also promising. Blockchain can provide a permanent, shareable, auditable record of products through their supply chain, which improves product traceability, authenticity, and legality in a more cost-effective way. In this chapter, the potential improvement expectations via blockchain technology for the case of agribusiness were discussed. The proposed case for automotive manufacturing-micro factory with blockchain technology was also introduced.

TITLE 7: Supply Chain Management based on Blockchain: A Systematic Mapping Study

AUTHORS: Youness Tribis

JOURNAL: MATEC Web of Conferences

YEAR: January 2018

DESCRIPTION: Groundbreakingly, blockchain technology (BCT) has gained widespread acceptance and importance in the last few years. Implemented in different areas of applications such as social and legal industries, finance, smart property, and supply chain networks. This technology assures immutability and integrity of data without the need of a third trusted party. Furthermore, BCT could guarantee a transparent and decentralized transaction system in businesses and industries. Even though general research has been done in the BCT, however, there is a lack of systematic analysis on current research challenges regarding how BCT is effectively applicable in supply chain management (SCM). A systematic literature review (SLR) of SCM based on blockchain does not exist yet. This work aims to explore and analyse the state-of-the-art on the BCT applications for SCM. We synthesize existing evidence, and identify gaps, available in the literature. The survey uses a systematic mapping study (SMS) method to examine 40 extracted primary studies from scientific databases.

TITLE 8: Role of Blockchain Technology in Digitization of Land Records in Indian Scenario

AUTHORS: P Singh

JOURNAL: IOP Conference Series: Earth and Environmental Science

YEAR: October 2020

DESCRIPTION: Blockchain is a way of passing data (such as records, events, or transactions) from one party to another in a very secure way. It is an electronic record of information that requires digital security. All data stored in the blockchain is immutable; once a piece of data enters into a blockchain, it is practically impossible to alter its value. The Blockchain has changed the model from a centralized way of traditional business to the decentralized model of the blockchain, that means there it can run without any central authority. It works on Peer to Peer Model rather than Peer-Mediator-Peer. The Blockchains makes the process easier, fastest, and trustworthy to deal with businesses as it follows Peer to Peer nature Blockchain. It has become the most used business model in different industries, such as construction industries, as it is the safest, fastest, transparent, and it also is more comfortable to implement. The critical feature of the blockchain that makes it today's most potential technologies are: decentralized, self-control, peer- to- peer relationship, fixed record and time stamping. Thus, this chapter focuses on the pivotal role and application of BlockChain technology on the digitization of land records of the Indian Scenario.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In the case study for product traceability , and the implemented system is named originChain. This system applies traceability of product by replacing normal centralized database with Blockchain data storage. The main idea of this system is to record the lab's product sample-testing results. A product ownership management system published in 2017 presents a system that implements Ethereum to provide the holding certificate of the consumer and combined the RFID of products to make sure that the product has its own identity stored in the Blockchain. On the business side, a company called Seal Network is combining Blockchain technology and Near-field communication (NFC) to develop a product authentication platform. This company inserts NFC chips into each item and use them as the certificates of the product. The NFC data is uploaded into the company's Blockchain.

Disadvantages

The proposed scheme can not guarantee that the product the consumer purchased from the seller is not a counterfeited one. Hence, the product counterfeit problem is still unsolved. Using NFC chips is not suitable for all types of products. For instance, fresh food or small commodities. Furthermore, in this kind of system, customers still get the products from the sellers and not directly from the manufacturer and reasonably the consumers may have concerns trusting the sellers

3.2 PROPOSED SYSTEM

The proposed system involves the use of a mobile application for the end user along with Blockchain technology in the backend as the base architecture ensure that the contents of its data are tampered proof. The mobile application will allow the user to scan the qr code on the product. On scanning, the app will display a list of the verified distributors through which the product has been. Since every distributor has to scan and transfer the ownership before selling to the next distributor. Hence the app tracks the authenticity of a product. Finally the end user can scan and verify the manufacturer and the last owner of the product which should be the seller. Hence, authenticity of product is ensured.

3.3 REQUIREMENT SPECIFICATION

3.3.1 Hardware Requirements

Processor	: Pentium Dual Core 2.00GHZ
Hard disk	: 120 GB
RAM	: 2GB (minimum)
Keyboard	: 110 keys enhanced

3.3.2 Software Requirements

Operating System	: Windows7 , 8, 8.1 and 10. Android OS
IDE	: Visual Studio Code, Remix
Language	: Solidity, Dart, Flutter
Other	: Ganache, Truffle

3.4 PLATFORM SPECIFICATION – GANACHE

First and foremost, Ganache is a component of the Truffle Suite ecosystem. The Truffle Suite includes Ganache as well as two additional tools: Truffle and Drizzle. Truffle is an EMV (Ethereum Virtual Machine) development environment, asset pipeline, and testing framework, whereas Drizzle is a collection of frontend libraries. Ganache, on the other hand, is a high-end development tool for running your own private blockchain for Ethereum and Corda dApp development. Ganache is useful throughout the developing process. The local chain provides a predictable and secure environment in which to build, deploy, and test your applications and smart contracts.

Ganache is available in two "versions," a desktop programme and a command-line tool. Ganache UI is a desktop programme that supports both Ethereum and Corda development; however, ganache-CLI is a command-line tool that only supports Ethereum development. Additionally, all of Ganache's versions are available for Mac, Windows, and Linux.

Ganache is used for two primary reasons: the first is to save money, and the second is to save time. When it comes to development, we strive to be as cost-effective and efficient as possible, which should be familiar to everyone who has used Moralis. Nonetheless, we must always pay gas costs when uploading smart contracts to the Ethereum network.

We may also use our local blockchain to rapidly upload our work. Uploading to Ethereum's main and testnets takes time, which might be a problem if you're attempting to develop quickly. This implies we'll have to wait for our contracts to be deployed on the blockchain before testing them, which can be avoided by using Ganache to set up a local blockchain.

How to Use Ganache

To download Ganache from the following URL –

<https://truffleframework.com/ganache>

Ganache is available on several platforms. We developed and tested this entire tutorial on Mac. Thus, the screenshots below will show Mac installation. When you open the installation URL given above, it automatically detects your machine's OS and directs you to the appropriate binary installation. The screenshot below shows the Mac installation.

Locate the “Ganache-2.0.0.dmg” in your Downloads folder and double-click on it to install Ganache. Drag Ganache icon to the Application folder. Now, Ganache is available as an application on your Mac. If you are using some other OS, follow the instructions provided for successful installation.

Now locate Ganache in your Application folder and double-click on its icon to start Ganache. Click QUICKSTART to start Ganache. The console in the above screenshot shows two user accounts with balance of 100 ETH (Ether - a currency for transaction on Ethereum platform). It also shows a transaction count of zero for each account. As the user has not performed any transactions so far, this count is obviously zero.

We will now get an overview of a few important screens of Ganache that are of immediate relevance to us.

How to Use Flutter

Flutter is basically Google's portable user interface (UI) toolkit, used to build and develop eye-catching, natively-built applications for mobile, desktop, and web, from a single codebase. Flutter is free, open-sourced, and compatible with existing code. It is utilized by companies and developers around the world, due to its user-friendly interface and fairly simple, yet to-the-point commands.

Flutter SDK is the tool that not only allows us to create flutter projects but also build those projects and transform them into native mobile applications. In simpler words, **Flutter SDK is the core tool for building a flutter UI.**

Once the zip file is downloaded, extract the '**flutter**' folder (drag and drop) to any path/directory of the system where you get the **read and write access**. Typically, it is better to create a new folder in a separate directory apart from the system drive due to permission issues (In my case, the target destination is **D: > development > flutter**).

Now double-click on the '**flutter**' folder. Go to '**flutter_console.bat**' file and double-click to open a command prompt window.

This console is actually a Windows terminal available for the developer to run flutter commands. Type in '**flutter**' to get a list of all the flutter commands that can be run. Check and edit environment variables for global system access. For this, scroll down to '**Update your path**' on the official Docs page of the flutter installation page. For this, go to **Control Panel > System and Security > System > Advanced System Settings > Environment Variables...** . A dialog box displaying a list of the available environment variables appears on your screen. Environment Variables are global system variables present at the root level, which aids in configuring various aspects of Windows. We will now add the flutter tool as an environment variable for direct access (instead of running the .bat executable), and unlock it on the entire PowerShell and Command Prompt of your system.

Installing Truffle in Flutter Project

Setting up the development environment

Truffle is the most popular development framework for Ethereum with a mission to make your life a whole lot easier. But before we install truffle make sure to install node .

Once we have node installed, we only need one command to install Truffle:

npm install -g truffle

Creating a Truffle Project

1. Create a basic Flutter project in your favorite IDE
2. Initialize Truffle in the flutter project directory by running

Truffle init

- **contracts/** : Contains solidity contract file.
- **migrations/**: Contains migration script files (Truffle uses a migration system to handle contract deployment).
- **test/** : Contains test script files.
- **truffle-config.js**: Contains truffle deployment configurations information.

Compiling and Migrating

In a terminal, make sure you are in the root of the directory that contains the flutter and truffle project, Run the following command: **truffle compile**

- Before we can migrate our contract to the blockchain, we need to have a blockchain running. For this article, we're going to use **Ganache**, a personal blockchain for Ethereum development you can use to deploy contracts, develop applications, and run tests. If you haven't already, download **Ganache** and double-click the icon to launch the application. This will generate a blockchain running locally on port 7545.
- Migrating the contract to the blockchain, run **truffle migrate**

Running the tests

Running the test as: **truffle test**

CHAPTER 4

SYSTEM DESIGN

4.1 ER DIAGRAM

ER stands for Entity Relationship. These diagrams display the relationship of entities that are used and stored in the database. They explain the structure of the whole process. these diagrams can be made using three basic concepts, attributed, relationships, and entities.

In Figure 4.1, the user has many required fields like email, address and password. User uses this fields to access the smart contract which is then connected with the asset. The assets contains fields like name , sha256 hash of QR, Identification details and owner address.

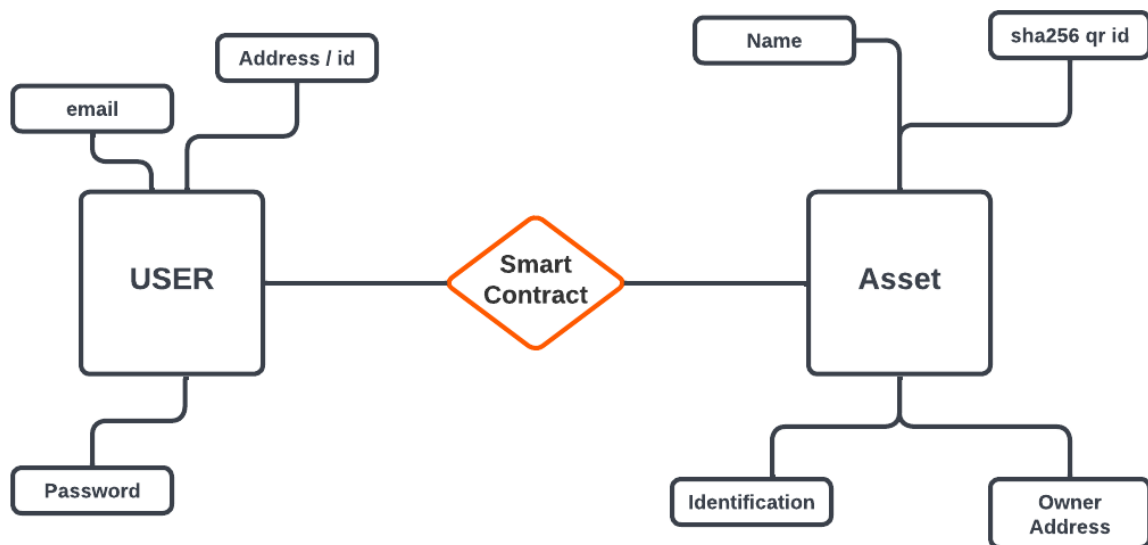


Fig 4.1 – ER Diagram

4.2 DATA FLOW DIAGRAM LEVEL 0

This is basically a contextual diagram, also referred to as a “context diagram”. It

only represents the top level or the 0 Level in the whole process. it gives an abstraction kind of view and shows the whole system as a single process and its relationship to externality.

In Figure 4.2, the user communicated with the smart contract which further calls the database. The database in return communicated back with smart contract.

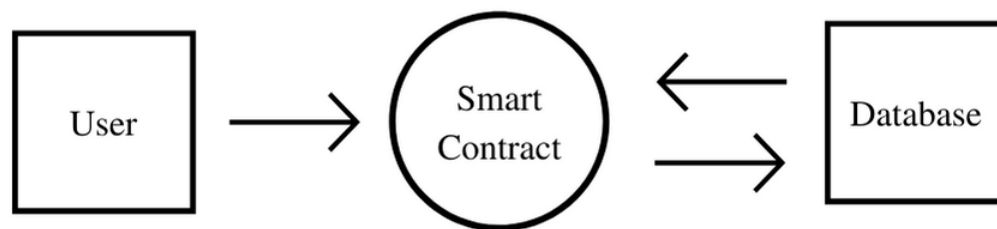


Fig 4.2 – Data Flow Diagram 0

4.3 UML DIAGRAM

4.3.1 Use Case

A UML use case diagram is the primary form of system/software requirements for a new software program underdeveloped.

In Figure 4.3.1, the admin first adds initial block, that is the first assigning of the ownership. Then a QR code is generated which the user and admin can use to view the product details and owner. The transferred ownership is sent to smart contract which verifies the hash and adds the data to the block.

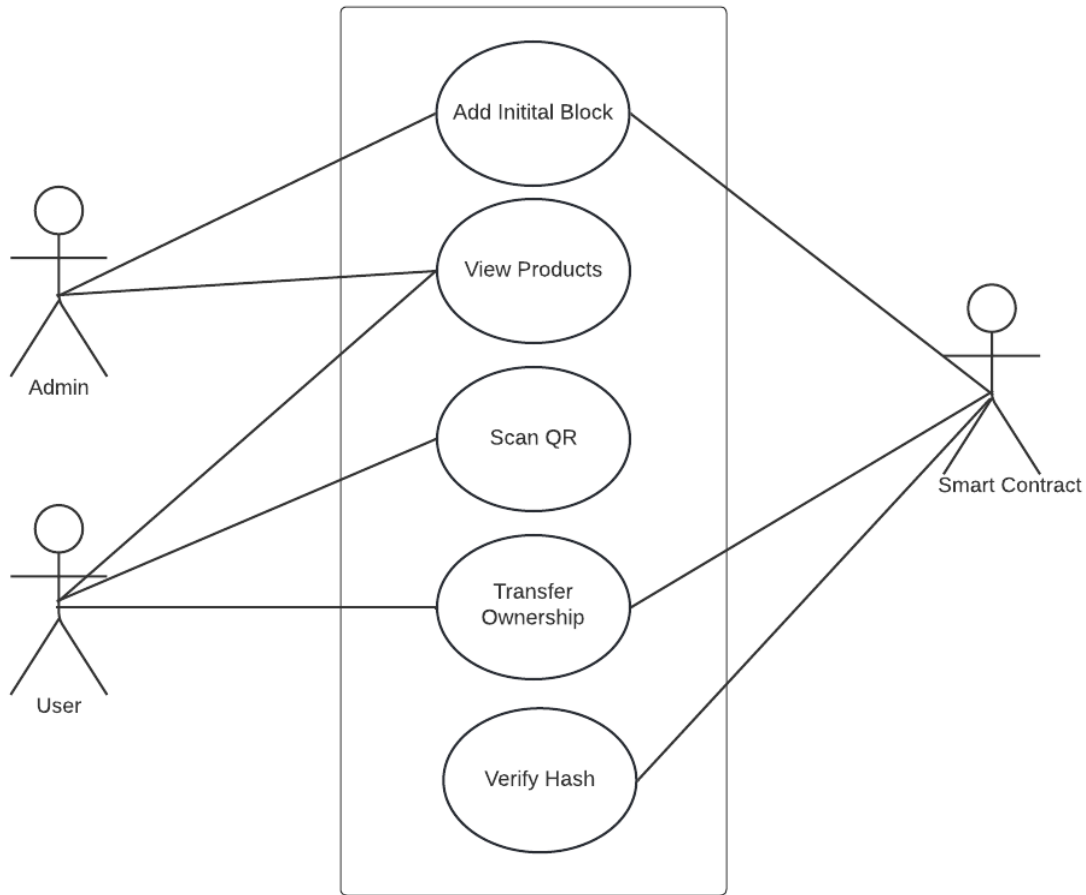


Fig 4.3.1 – Use Case

4.3.2 Activity Diagram

In simple terms, a diagram that represents the order of all activities is called the activity diagram. It shows the workflow between different activities that take place in the whole process. However, these are not exactly flowcharts but are similar.

In Figure 4.3.2, the activity flow from the user is followed with the scanning of qr. Then it is verified in the smart contract which gives two values. Valid or invalid, if valid the user has a option of transferring the ownership to other user. If yes, then the data is passed to smart contract which results in the final output.

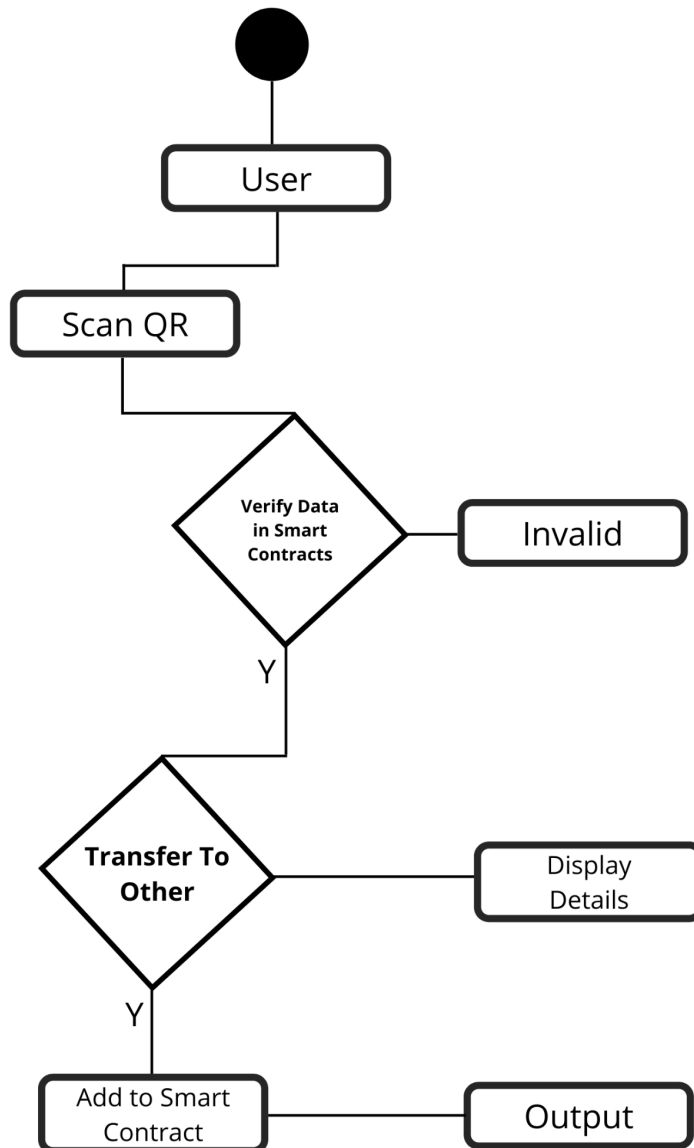


Fig 4.3.2 – Activity Diagram

4.3.3 Sequence Diagram

A sequence diagram shows the sequence of message passed between objects. In Figure 4.3.3, the sequence flows as the user sends the input to the application. In the application the user verification happens, product ownership verification which is followed by the smart contract. It final output is sent back to the user.

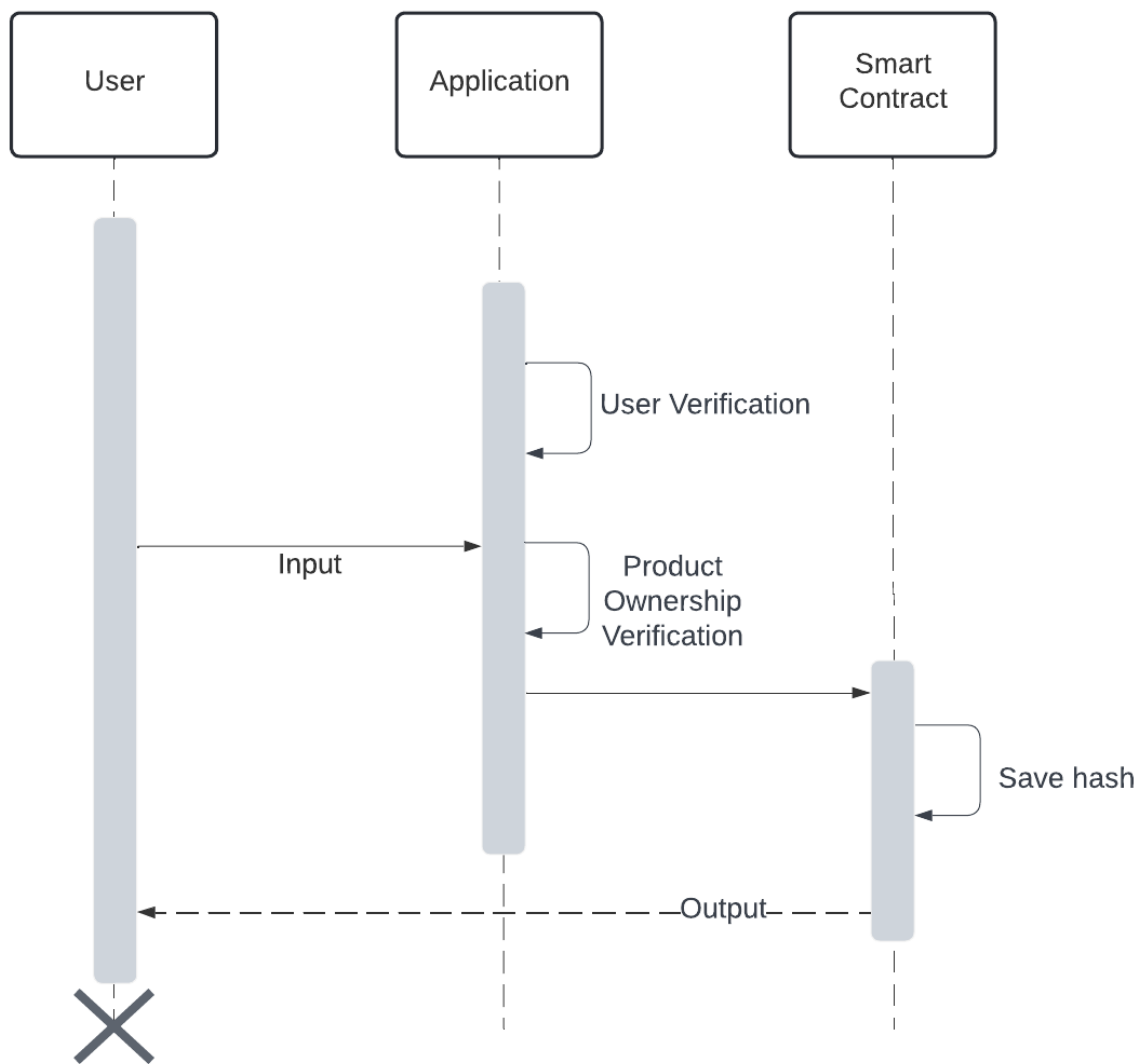


Fig 4.3.3 –Sequence Diagram

4.3.4 Class Diagram

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system.

In Figure 4.3.4, the admin contains datas like address and license, which is connected with user which has address,email and password. Product Details has name, image in binary form, address that needs to be transferred and other details

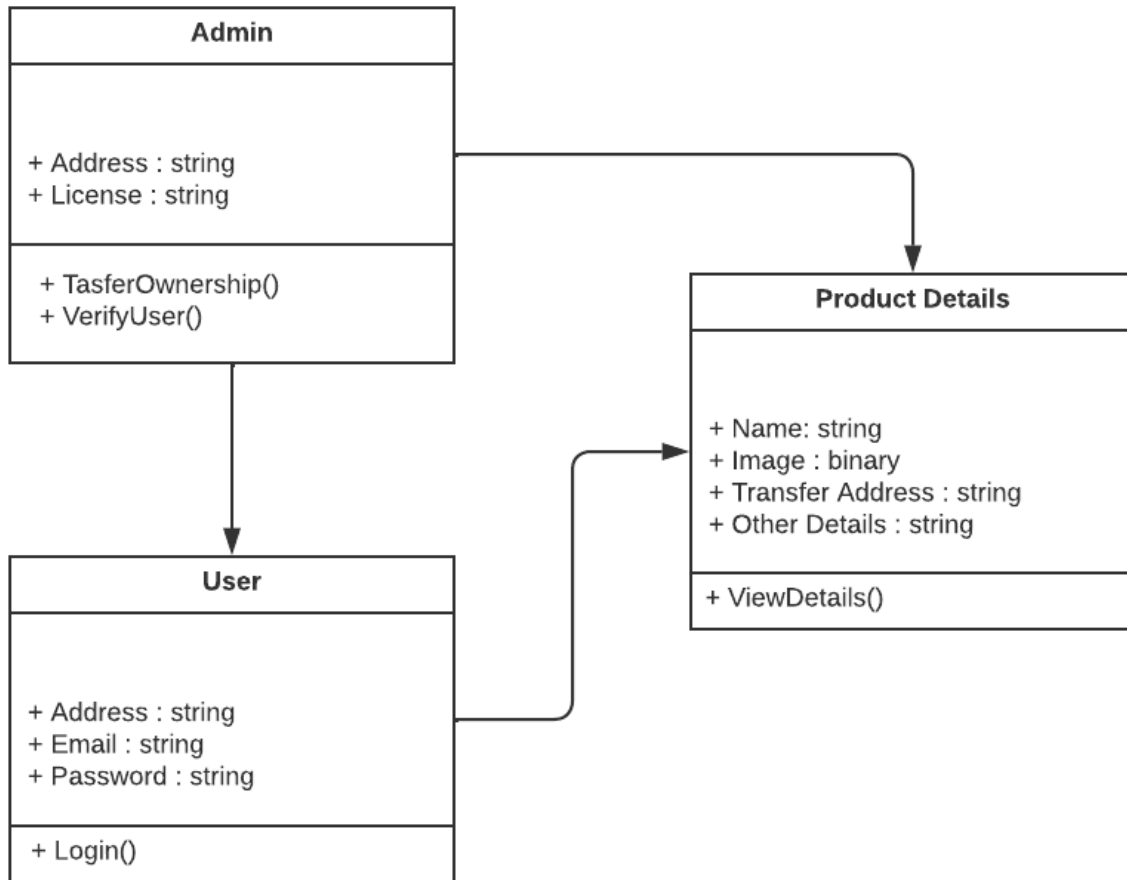


Fig 4.3.4 – Class Diagram

CHAPTER 5

MODULE DESCRIPTION

5.1 CREATION OF SMART CONTRACT

Since a smart contract acts as a single source of automated verification of the blocks, it is to be created first. Using a programming language called Solidity and an editor called Remix this can be done. Remix provides a stimulated blockchain environment for testing purposes with multiple accounts and 100 Ethers each.

5.2 INTEGRATING WITH MOBILE APPLICATION

5.2.1 Flutter

Blockchain is a backend process, but to let the user interact with blockchain we need a user interface. In this case we create a mobile app using a google's framework called Flutter. After successful installation, a flutter code is written to create the ui of the app which then will be accessed by the smart contract.

5.2.2 Migration of Smart contract in App

Third party tools like TRUFFLE is installed using node package manager which will provide us a blockchain stimulation for our mobile app testing. Along with that we install a software called Ganache where we can access all the key,accounts etc

Once adding the solidity code in app project level file, try to compile it locally using **truffle compile**.

Then connect solidity function with the application ui for a streamlined communication. Once the transaction takes place it can be seen in ganache for reference. We can see some ether being reduced from a particular account. It is nothing but a gas fee which mean the transaction was successful in blockchain.

ACCOUNTS	BLOCKS	TRANSACTIONS	CONTRACTS	EVENTS	LOGS	SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 5	GAS PRICE 2000000000	GAS LIMIT 6721975	HARDFORK MUIRGLACIER	NETWORK ID 5777	RPC SERVER HTTP://192.168.1.243:7545	MINING STATUS AUTOMINING
WORKSPACE MINOR-DESTRUCTION						SWITCH
TX HASH 0x55d56195e885be61a86c88a824c0630e99eb797de499c941836f6fa634924167						CONTRACT CALL
FROM ADDRESS 0x95a898AEC8550Dd06f2c2EBfE478D65789F4B8e1		TO CONTRACT ADDRESS 0x752C79aDBf644ed40553e6d8F2a83EB913e2D760		GAS USED 67011	VALUE 0	
TX HASH 0x4efd4b341eb3a166dbca4b14a345cfe2dab8cf436d8d241c3a6a16f3a1b005f8						CONTRACT CALL
FROM ADDRESS 0x95a898AEC8550Dd06f2c2EBfE478D65789F4B8e1		TO CONTRACT ADDRESS 0xbE03b1d0A32D75Cd5B7353f42C142e758aF7828A		GAS USED 27513	VALUE 0	
TX HASH 0xe2318530758cdb399bf2f867fa57f1dbc9fc1478db35a9bd1a78a3c36a2a0bbd						CONTRACT CREATION
FROM ADDRESS 0x95a898AEC8550Dd06f2c2EBfE478D65789F4B8e1		CREATED CONTRACT ADDRESS 0x752C79aDBf644ed40553e6d8F2a83EB913e2D760		GAS USED 566507	VALUE 0	
TX HASH 0x91ba644e7398983744e16a665c781b0de7c55900861bafb866107eb51ff94e4a						CONTRACT CALL
FROM ADDRESS 0x95a898AEC8550Dd06f2c2EBfE478D65789F4B8e1		TO CONTRACT ADDRESS 0xbE03b1d0A32D75Cd5B7353f42C142e758aF7828A		GAS USED 42513	VALUE 0	
TX HASH 0xf4308a4235e6b7c286a9ca4bc2b2c303a7a407d3539749ccee203a255ea8235						CONTRACT CREATION
FROM ADDRESS 0x95a898AEC8550Dd06f2c2EBfE478D65789F4B8e1		CREATED CONTRACT ADDRESS 0xbE03b1d0A32D75Cd5B7353f42C142e758aF7828A		GAS USED 248854	VALUE 0	

Fig 5.2.2 Ganache

5.3 ASSIGNING OF OWNERSHIP

From the app the admin can add the product details and assign the ownership to a particular address. First the data is hashed and stored in smart contract for the respective address. Then the remaining data is stored in database for additional data information of the product. Products is the hashed key which is assigned to the owner address id.

5.4 OUTPUT

The final step of our proposed system is to assign the ownership of the asset which is successfully done. The owner can then transfer the asset to some other user as well. To view the product details a qr scanner has also been added. Thus our aim to solve asset anti-counterfeiting has been successfully done.

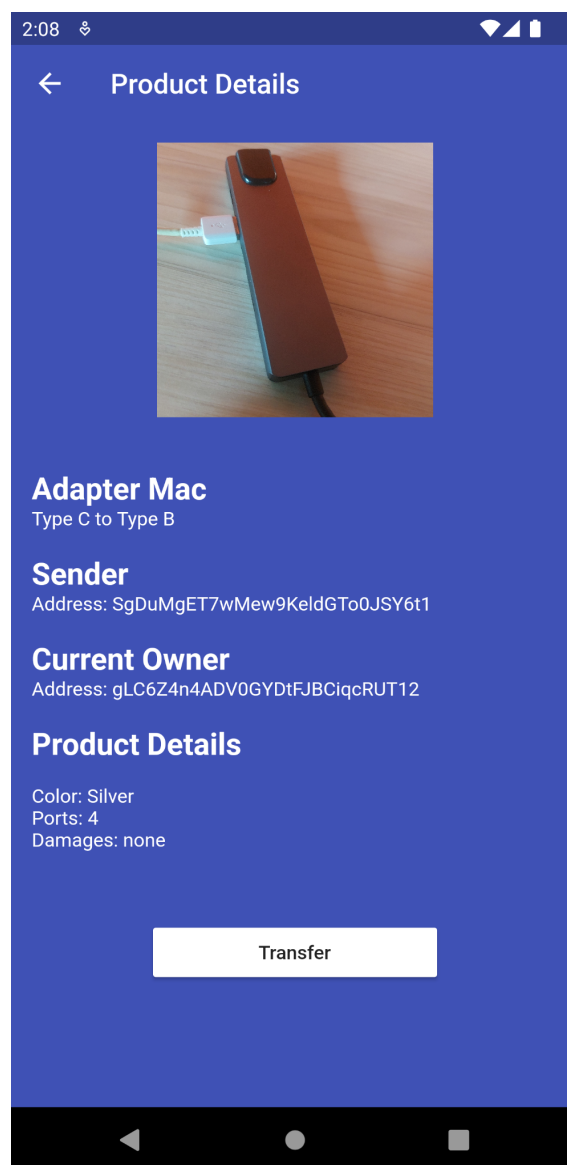


Fig 5.4 Application

CHAPTER 6

TESTING


S.NO	Action	Inputs	Expected Output	Actual Output	Test Result
1	Admin Login	hussain@admin.com	Response 200	Response 200	Pass
2	User Login	kumar@gmail.com	Response 200	Response 200	Pass
3	Add Product	Product Details	Item Added	Item Added	Pass
4	Scan Qr	 <p>Adapter Mac</p>	Product Details	Product Details	Pass

Table 6.1

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

Thus an Blockchain based product anti counterfeiting application was successfully made. Following extensive testing, comparing results revealed that the model generates expected results which is an important strategy for application in open contexts. The application can be used in other business use case like supply chain management. With the future expansion of the model it can be more efficient. Future expansion of the system could include adding Bluetooth low energy devices to the product and an NFT based product identification software. Each asset can be assigned with a Bluetooth low energy device that would emit a radio wave which can be identified by the mobile application. That would further improve the authenticity and transparency and would help to ease the process of supply chain and management. Since the current model works on the QR code it can be further transformed to NFT for assets where bluetooth low energy can be used.

CHAPTER 8

SYSTEM IMPLEMENTATION

6.1 Client-side coding

Connecting Smart contract with App

```
import 'dart:convert';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:http/http.dart';
import 'package:web3dart/web3dart.dart';
import 'package:web_socket_channel/io.dart';

class Model extends ChangeNotifier {
  bool isLoading = true;
  int taskCount = 0;

  final String _rpcURL = "http://192.168.1.243:7545";
  final String _wsURL = "ws://192.168.1.243:7545/";

  final String _privatekey =
    "81142a9c3f54d9e1f7e7d7a283a7105482acacc6429e6bb8a6e70b551e636230";

  late final Web3Client _client;

  String? _abiCode;

  EthereumAddress? _contractAddress;

  Credentials? _credential;
```

```
EthereumAddress? _ownaddress;
```

```
DeployedContract? _contract;
```

```
ContractFunction? _addProductOwnership;
```

```
ContractFunction? _getProductDetails;
```

```
ContractFunction? _transferOwnership;
```

```
ContractEvent? _taskCreatedEvent;
```

```
Model() {  
    initiateSetup();  
}
```

```
Future<void> initiateSetup() async {  
    _client = Web3Client(  
        _rpcURL,  
        Client(),  
        socketConnector: () {  
            return IOWebSocketChannel.connect(_wsURL).cast<String>();  
        },  
    );  
    await getAbi();  
    await getCredentials();  
    await getDeployedContract();  
}
```

```
Future<void> getAbi() async {  
    String getabistring =  
        await rootBundle.loadString('build/contracts/Genie.json');  
    var jsonAbi = jsonDecode(getabistring);
```

```

    _abiCode = jsonEncode(jsonAbi['abi']);

    _contractAddress =
        EthereumAddress.fromHex(jsonAbi['networks']['5777']['address']);
    print(_contractAddress);
}

Future<void> getCredentials() async {
    _credential = await EthPrivateKey.fromHex(_privatekey);
    _ownaddress = await _credential!.extractAddress();
}

Future<void> getDeployedContract() async {
    print('end');

    _contract = DeployedContract(
        ContractAbi.fromJson(_abiCode!, "Genie"),
        _contractAddress!,
    );
    print('end2');

    _addProductOwnership = await _contract!.function("addProductOwnership");
    print('end3');

    _getProductDetails = await _contract!.function("getProductDetails");
    print('end4');

    _transferOwnership = await _contract!.function("transferOwnership");
    print('end5');
}

```

```

addProdOwnership(String qr, String uid) async {
  await _client.sendTransaction(
    _credential!,
    Transaction.callContract(
      contract: _contract!,
      function: _addProductOwnership!,
      parameters: [qr, uid, _ownaddress]));
  print('success');
}
}

```

Flutter Mobile Application

Initial transfer from admin

```

import 'dart:convert';
import 'dart:io';

import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:crypto/crypto.dart';

import 'package:genie/DB/db.dart';
import 'package:genie/screens/qrdisplay.dart';
import 'package:image_picker/image_picker.dart';
import 'package:provider/provider.dart';

import '../model.dart';

class TransferAdmin extends StatefulWidget {

```

```

@override
State<TransferAdmin> createState() => _TransferAdminState();
}

```

```

class _TransferAdminState extends State<TransferAdmin> {
  List<String>? additionalDetails;

```

```

  TextEditingController prodName = TextEditingController();

```

```

  TextEditingController prodIden = TextEditingController();

```

```

  TextEditingController address = TextEditingController();

```

```

  File? _image;

```

```

  File? _imageFull;

```

```

  Future<File?> getImageFromGallery() async {

```

```

    final picker = ImagePicker();

```

```

    final pickedFile = await picker.getImage(

```

```

      source: ImageSource.gallery,

```

```

    );

```

```

    if (pickedFile != null) {

```

```

      return File(pickedFile.path);

```

```

    } else {

```

```

      //print('No image selected.');
```

```

    }

```

```

    return null;

```

```

  }

```

```

@override

```

```

Widget build(BuildContext context) {

```

```

  var listmodel = Provider.of<Model>(context);

```

```

final FirebaseAuth auth = FirebaseAuth.instance;

var height = MediaQuery.of(context).size.height - kToolbarHeight;
return Material(
  child: Scaffold(
    backgroundColor: Theme.of(context).primaryColor,
    appBar: AppBar(
      elevation: 0,
      title: const Text('Transfer Ownership'),
      actions: [
        Container(
          margin: EdgeInsets.all(12),
          child: MaterialButton(
            color: Colors.white,
            child: Text('Transfer'),
            onPressed: () async {
              print(prodName.text);
              print(prodIden.text);
              print(address.text);
              print(additionalDetails);

              ///
              final User? user = auth.currentUser;
              final uid = user!.uid;

              ///
              String other = "";
              for (var item in additionalDetails!) {
                other = other + '\n' + item;
              }
            }
          )
        )
      ],
    ),
  ),
);

```

```

///
var appleInBytes = utf8.encode("${prodName.text}-");
Digest qr = sha256.convert(appleInBytes);
print(qr.toString());

///
await DB().upload(
  admin_uid: uid,
  image: _image!,
  qr: qr.toString(),
  prodname: prodName.text,
  identification: prodIden.text,
  transfer_Address: address.text,
  otherDetails: other,
);

Navigator.of(context).pop();
Navigator.push(
  context,
  MaterialPageRoute(
    builder: (BuildContext context) => QRdisplay(
      data: qr.toString(),
      name: prodName.text,
    ),
  ),
);
//
print('adding to blockchain');
listmodel.addProdOwnership(
  qr.toString(), address.text.trim());
},

```



```

    ),
  ),
],
),
body: SingleChildScrollView(
  child: Container(
    margin: const EdgeInsets.all(15),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,
      children: [
        _image != null
          ? Center(
              child: Image.file(
                _image!,
                fit: BoxFit.cover,
                height: height * 0.2,
              ),
            )
        : InkWell(
            onTap: () async {
              File? temp = await getImageFromGallery();
              setState(() {
                _image = temp;
              });
            },
            child: Container(
              width: double.infinity,
              height: height * 0.28,
              decoration: BoxDecoration(
                border: Border.all(color: Colors.grey),
                borderRadius: BorderRadius.circular(12),

```

```
),  
    child: Center(  
      child: Icon(Icons.camera_alt_rounded),  
    ),  
  ),  
),  
  
SizedBox(height: 40),  
Text('Product Name', style: TextStyle(color: Colors.white)),  
SizedBox(height: 8),  
TextField(  
  controller: prodName,  
  decoration: InputDecoration(  
    fillColor: Colors.white,  
    border: OutlineInputBorder(),  
    enabledBorder: OutlineInputBorder(  
      borderSide: BorderSide(color: Colors.white, width: 1.0),  
    ),  
    focusedBorder: OutlineInputBorder(  
      borderSide: BorderSide(color: Colors.white, width: 1.0),  
    ),  
  ),  
),  
  
SizedBox(height: 20),  
  
///  
///  
///  
  
Text('Product Identification',  
  style: TextStyle(color: Colors.white)),  
SizedBox(height: 8),  
TextField(
```

```
controller: prodIden,  
decoration: InputDecoration(  
  border: OutlineInputBorder(),  
  enabledBorder: OutlineInputBorder(  
    borderSide: BorderSide(color: Colors.white, width: 1.0),  
  ),  
  focusedBorder: OutlineInputBorder(  
    borderSide: BorderSide(color: Colors.white, width: 1.0),  
  ),  
),  
),  
),  
SizedBox(height: 40),  
  
///  
///  
///  
Text('Transfer Address', style: TextStyle(color: Colors.white)),  
SizedBox(height: 8),  
TextField(  
  controller: address,  
  decoration: InputDecoration(  
    border: OutlineInputBorder(),  
    enabledBorder: OutlineInputBorder(  
      borderSide: BorderSide(color: Colors.white, width: 1.0),  
    ),  
    focusedBorder: OutlineInputBorder(  
      borderSide: BorderSide(color: Colors.white, width: 1.0),  
    ),  
  ),  
),  
),  
),  
SizedBox(height: 40),
```

```

    ///
    ///
    ///
    Text('Additional Product Details',
        style: TextStyle(color: Colors.white)),
    SizedBox(height: 8),
    MultipleFields(
        totalAns: ((allAns) => additionalDetails = allAns),
    )
],
),
),
),
),
);
}
}

```

```

class MultipleFields extends StatefulWidget {
    final void Function(List<String> allAns)? totalAns;
    MultipleFields({this.totalAns});
    @override
    State<MultipleFields> createState() => _MultipleFieldsState();
}

```

```

class _MultipleFieldsState extends State<MultipleFields> {
    int textfieldcount = 2;
    List<String> ans = [ "", "" ];
    @override
    Widget build(BuildContext context) {

```

```

return Column(
  children: [
    ListView.builder(
      shrinkWrap: true,
      physics: NeverScrollableScrollPhysics(),
      itemCount: textfieldcount,
      itemBuilder: (ctx, i) {
        return Column(
          children: [
            TextField(
              decoration: const InputDecoration(
                border: OutlineInputBorder(),
                enabledBorder: OutlineInputBorder(
                  borderSide: BorderSide(color: Colors.white, width: 1.0),
                ),
                focusedBorder: OutlineInputBorder(
                  borderSide: BorderSide(color: Colors.white, width: 1.0),
                ),
              ),
              onChanged: (val) {
                ans[i] = val;

                widget.totalAns!(ans);
              },
            ),
            SizedBox(height: 20),
          ],
        );
      },
    ),
    FlatButton(

```

```

onPressed: () {
  setState(() {
    textfieldcount = textfieldcount + 1;
    ans.add("");
  });
},
child: Text(
  'Add',
  style: TextStyle(color: Colors.white54),
),
),
],
);
}
}

```

Product Details

```

import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:genie/DB/db.dart';
import 'package:genie/screens/user/prodModel.dart';

```

```

import '../constants/textStyleconst.dart';

```

```

class ProductDetails extends StatelessWidget {
  ProdModel product;
  bool canTransfer;
  ProductDetails({
    required this.product,
    required this.canTransfer,

```

```

});
@override
Widget build(BuildContext context) {
  TextEditingController trasferTo = TextEditingController();
  return Scaffold(
    backgroundColor: Theme.of(context).primaryColor,
    appBar: AppBar(
      elevation: 0,
      title: Text('Product Details'),
    ),
    body: SingleChildScrollView(
      child: Container(
        margin: EdgeInsets.all(15),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.start,
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Center(
              child: Image.network(
                '${product.image}',
                height: 200,
                width: 200,
              ),
            ),
            SizedBox(height: 40),
            Text('${product.name}', style: h1),
            Text('${product.identification}', style: wc),
            SizedBox(height: 20),
            Container(
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,

```

```

        children: [
          Text('Sender', style: h1),
          Text('Address: ${product.senderAddress}', style: wc),
        ],
      ),
    ),
    SizedBox(height: 20),
    Container(
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text(
            'Current Owner',
            style: h1,
          ),
          Text('Address: ${product.transferAddress}', style: wc),
        ],
      ),
    ),
    ),
    SizedBox(height: 20),
    Container(
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text('Product Details', style: h1),
          Text(
            '${product.otherDetails}',
            style: wc,
          ),
        ],
      ),
    ),
  ),

```



```

),
SizedBox(height: 50),
if (canTransfer)
Center(
  child: MaterialButton(
    child: Text('Transfer'),
    color: Colors.white,
    minWidth: MediaQuery.of(context).size.width / 2,
    onPressed: () {
      showDialog(
        context: context,
        builder: (BuildContext context) {
          return Dialog(
            shape: RoundedRectangleBorder(
              borderRadius: BorderRadius.circular(
                5.0)), //this right here
            child: Container(
              height: 300,
              child: Padding(
                padding: const EdgeInsets.all(12.0),
                child: Column(
                  mainAxisAlignment: MainAxisAlignment.center,
                  crossAxisAlignment:
                    CrossAxisAlignment.center,
                  children: [
                    Text(
                      'Transfer to',
                      style: TextStyle(
                        fontWeight: FontWeight.bold),
                    ),
                    SizedBox(height: 30),

```

[illegible]

```

        ///
        await DB().transferTo(
            uid,
            transferTo.text.trim(),
            product);
        print('2');

        Navigator.of(context).pop();
    },
    child: Text(
        "Transfer",
        style: TextStyle(
            color: Colors.white),
    ),
    color: const Color(0xFF1BC0C5),
    ),
    ),
    )
    ],
    ),
    ),
    ),
    );
    });
    },
    ),
    ),
    ],
    ),
    ),
    ),
    ),

```

```
);  
}  
}
```

6.2 Server-side coding

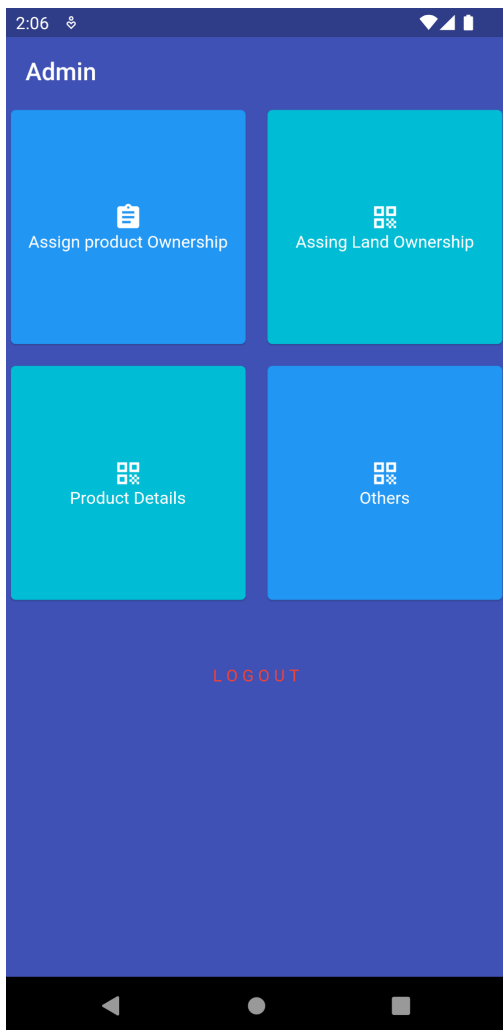
Genie.sol

```
// SPDX-License-Identifier: MIT  
pragma solidity ^0.8.1;  
  
contract Genie{  
    uint public taskCount;  
  
    struct User{  
        address adres;  
        string uid;  
    }  
    mapping(string => User) public ownership;  
  
    constructor(){  
  
    }  
  
    function addProductOwnership(string memory _productQR,  
    string memory _uid,  
    address _address) public{  
        ownership[_productQR] = User(_address,_uid);  
    }  
}
```

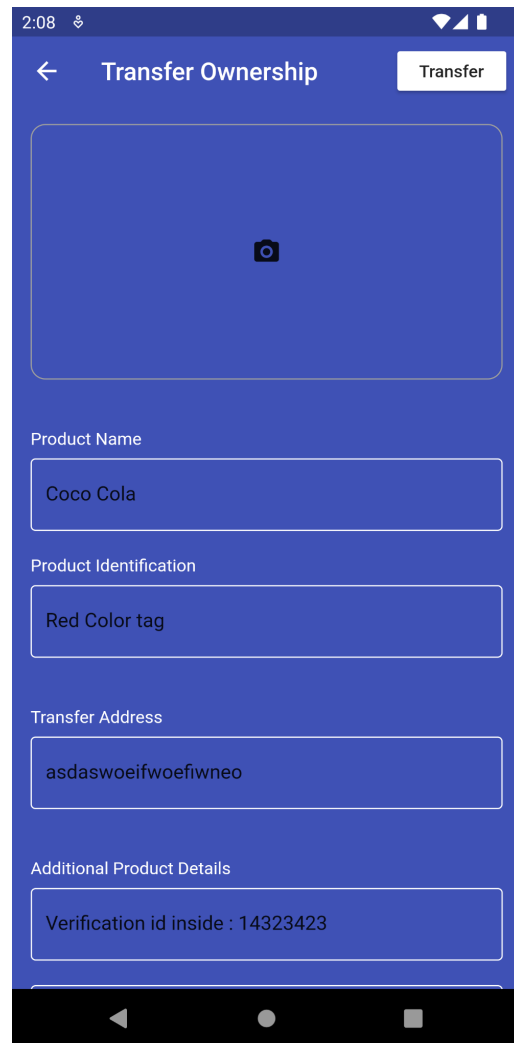
```
function getProductDetails(string memory _productQR)public view  
returns(string memory) {  
    return ownership[_productQR].uid;  
}
```

```
function transferOwnership(string memory _productQR,string memory  
_uid,address _address)public{  
    ownership[_productQR] = User(_address,_uid);  
}  
}
```

APPENDICES

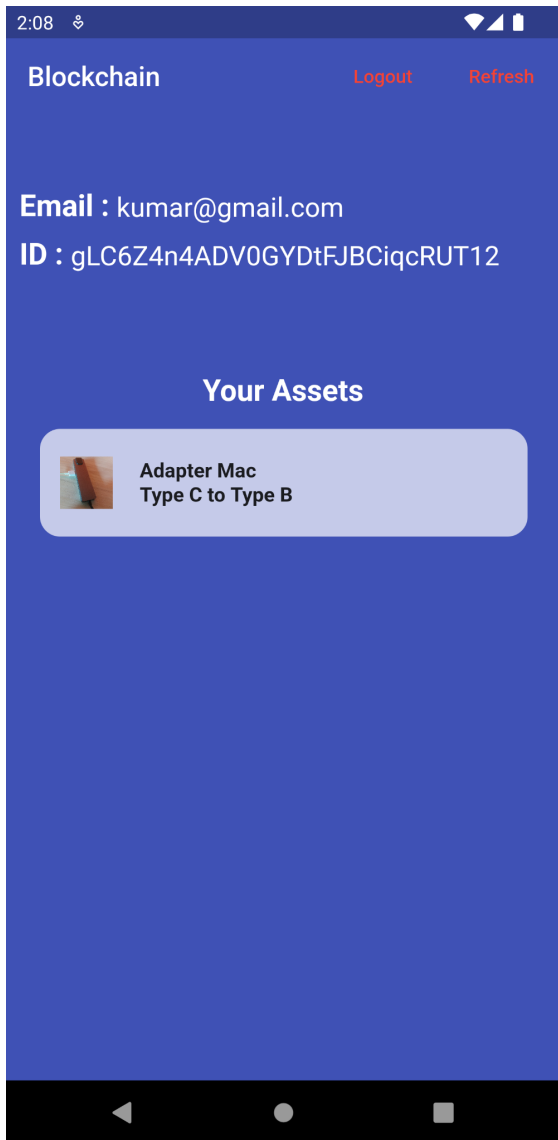


Home Screen of Admin App

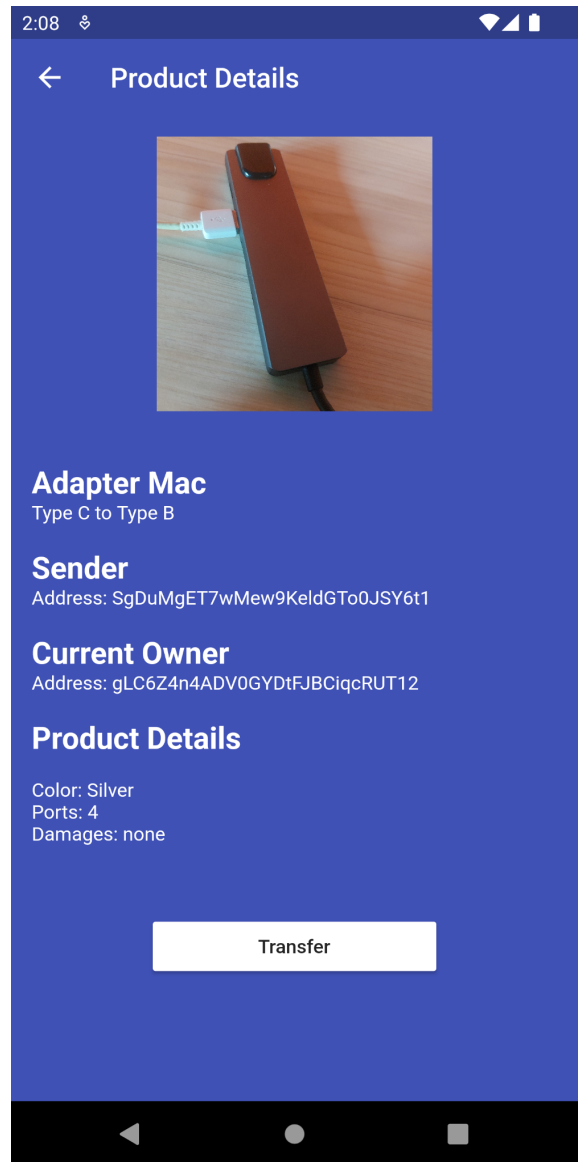


Screen to enter product details

Fig 9.1 Admin App



Home Screen of User App



Product Details Page

Fig 9.2 User App

REFERENCES

IEEE Paper

- [1] D. Puthal, N. Malik, S. P. Mohanty, E. Kougianos, and G. Das, "EVERYTHING YOU WANTED TO KNOW ABOUT THE BLOCKCHAIN", IEEE Consumer Electronics Magazine, Volume 7, Issue 4, July 2018, pp. 06–14.
- [2] S. Wang, L. Ouyang, Y. Yuan, X. Ni, X. Han and F. Wang, "Blockchain-Enabled Smart Contracts: Architecture, Applications, and Future Trends," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 49, no. 11, pp. 2266-2277, Nov. 2019

Web References

- [1] "QR Code features". Denso-Wave, Retrieved from <https://www.qrcode.com/en/>
- [2] Satoshi Nakamoto, 'BITCOIN: A PEER-TO-PEER ELECTRONIC CASH SYSTEM' (2009), Retrieved from www.bitcoin.org
- [3] Kessler G. C. (2019), 'AN OVERVIEW OF CRYPTOGRAPHY', Retrieved from <https://commons.erau.edu/publication/412>
- [4] "What are smart contracts on blockchain?", Retrieved from <https://www.ibm.com/in-en/topics/smart-contracts>

[5] “INTRODUCTION TO SMART CONTRACTS” (2021) , Retrieved from <https://ethereum.org/en/developers/docs/smart-contracts/>

[6] Eduard Daoud, Dang Vu, Hung Nguyen and Martin Gaedke,
IMPROVING FAKE PRODUCT DETECTION USING AI-BASED
TECHNOLOGY, 18th
International Conference e-Society 2020

[7] Duy-Thinh Tran, Sung Je Hong, ‘RFID Anti-Counterfeiting for Retailing Systems’, Journal of Applied Mathematics and Physics, 2015, 3, 1-9

[8] De Li, XueLi Gao, Yu Chao Sun and LiHua Cui ‘Research on Anti-counterfeiting Technology Based on QR Code image Watermarking Algorithm’
International Journal of Multimedia and Ubiquitous Engineering, ISSN: 1975-0080 IJMUE, Vol.12, No.5 (2017), pp.57-66

[9] Nader Dabit (2021), “What is Web3? The Decentralized Internet of the Future Explained”, Retrieved from <https://www.freecodecamp.org/news/what-is-web3/>

[10] Carson, B., Romanelli, G., Walsh, P. and Zhumaev, A., 2018. “Blockchain beyond the hype: What is the strategic business value”. McKinsey & Company,

[11] “Making sense of bitcoin, cryptocurrency and blockchain”, Retrieved from <https://www.pwc.com/us/en/industries/financial-services/fintech/bitcoin-blockchain-cryptocurrency.html>

Book References

[1] Arul Lawrence Sivakumar, C. Suresh Ganadhas, 'THE EVALUATION REPORT OF SHA-256 CRYPT ANALYSIS HASH FUNCTION', 2009
International
Conference on Communication Software and Networks

[2] Alharby, Maher & Aldweesh, Amjad & van Moorsel, Aad. (2019).
Blockchain-based Smart Contracts: A Systematic Mapping Study of Academic
Research
(2018).

[3] Dannen, Chris. (2017), 'Introducing Ethereum and Solidity'