COUNTERFEIT PRODUCT DETECTION USING BLOCKCHAIN

A Project Report
Submitted in partial fulfilment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
INFORMATION TECHNOLOGY

Submitted by

JAVVAJI HARINI 19BQ1A1256
GANJI PUSHPALATHA 19BQ1A1236
GOVATHOTI LAKSHMI TEJASWI 19BQ1A1243
ENUKOLLU DEDIPYA 19BQ1A1233

Under the Supervision of
Dr. ALLA KALAVATHI, M. Tech, Ph. D

Professor & HOD

DEPARTMENT OF INFORMATION TECHNOLOGY

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

NAMBUR(V), PEDAKAKANI(M), GUNTUR-522508, TEL NO:0873 211803,

url: www.vvitguntur.com An Autonomous Institute Affiliated to JNTUK, Approved by AICTE New Delhi, Accredited by NBA, NAAC with 'A' Grade and ISO 9001:2008 Certified

April 2023

VASIREDDY VENKATADRI INSTITUTE OFTECHNOLOGY: : NAMBUR

BONAFIDE CERTIFICATE

This is to certify that the project report **"COUNTERFEIT PRODUCT DETECTION USING BLOCKCHAIN"** is the bonafide work done by "**JAVVAJI HARINI (19BQ1A1256), GANJI PUSHPALATHA (19BQ1A1236), GOVATHOTI LAKSHMI TEJASWI (19BQ1A1243), ENUKOLLU DEDIPYA (19BQ1A1233)"**, who carried out the project under my guidance during the year 2023 towards partial fulfillment of the requirements of the Degree of Bachelor of Technology in Information Technology from Vasireddy Venkatadri Institute of Technology, Nambur. The results embodied in this report have not been submitted to any other University for the award of any degree.

Signature of the Head of the Department

Dr. ALLA KALAVATHI
HEAD OF THE DEPARTMENT
Department of Information Technology

Signature of the Supervisor

Dr. ALLA KALAVATHI
PROJECT GUIDE

External Viva voce conducted on 15-04-2023

Internal Examiner **External Examiner**

VASIREDDY VENKATADRI INSTITUTEOF TECHNOLOGY:: NAMBUR

CERTIFICATE OF AUTHENTICATION

I solemnly declare that this project report **"COUNTERFEIT PRODUCT DETECTION USING BLOCKCHAIN"** is the bonafide work done purely by me/us, carried out under the supervision of Dr. ALLA KALAVATHI, towards partial fulfillment of the requirements of the Degree of Bachelor of Technology in Information Technology from Vasireddy Venkatadri Institute of Technology, Nambur during the year 2022-23.

It is further certified that this work has not been submitted, either in part of in full, to any other department of the Vasireddy Venkatadri Institute of Technology, or any other University, institution or elsewhere, or for publication in any form.

Signature of the Student

15-04-2023

JAVVAJI HARINI
19BQ1A1256, 2022-23

GANJI PUSHPALATHA
19BQ1A1236, 2022-23

GOVATHOTI LAKSHMI TEJASWI
19BQ1A1243, 2022-23

ENUKOLLU DEDIPYA
19BQ1A1233 2022-23

ACKNOWLEDGEMENT

We take this opportunity to express our deepest gratitude and appreciation to all those people who made this project work easier with words of encouragement, motivation, discipline, and faith by offering different places to look to expand my ideas and helped me towards the successful completion of this project work.

First and foremost, we express our deep gratitude to **Sri. Vasireddy Vidya Sagar, Chairman,** Vasireddy Venkatadri Institute of Technology for providing necessary facilities throughout the Information Technology program.

We express our sincere thanks to **Dr. Y. Mallikarjuna Reddy, Principal,** Vasireddy Venkatadri Institute of Technology for his constant support and cooperation throughout the Information Technology program.

We express our sincere gratitude to **Dr. A. Kalavathi, Professor & HOD,** Information Technology, Vasireddy Venkatadri Institute of Technology for her constant encouragement, motivation and faith by offering different places to look to expand my ideas. We would like to express our sincere gratefulness to our guide **Dr. A. Kalavathi, Professor** for his/her insightful advice, motivating suggestions, invaluable guidance, help and support in successful completion of this project and also our project co-ordinator **Mr. R. Sudha Kishore, Associate Professor** for her advice and support. We would like to take this opportunity to express our thanks to the teaching and nonteaching staff in Department of Information Technology, VVIT for their invaluable help and support.

ABSTRACT

The primary goal of this project is to utilize blockchain technology to identify counterfeit products, which are essentially inferior copies of authentic brands. The purpose is to enhance the ability to detect fake products by using blockchain as a secure and protected mechanism for recording transactions. To prevent the duplication of products, a blockchain-based system can be employed. This system keeps a record of the supply chain of products at each stage of the transaction to a new party, thereby preserving the product's ownership history through a QR code. Consequently, when customers purchase the product, they can access complete information about its journey from the manufacturer to the retailer, ensuring that they acquire the original product. Overall, the aim of this project is to enhance the detection of counterfeit products and promote consumer trust in the supply chain, which can ultimately lead to increased customer confidence and satisfaction.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

QR	-	Quick Response
JSX	-	JavaScript eXtensible Markup Language
RAM	-	Random Access Memory
UML	-	Unified Modelling Language
CSS	-	Cascading Style Sheets
EVM	-	Ethereum Virtual Machine
JS	-	JavaScript

CHAPTER 1
INTRODUCTION

1.1 ABOUT THE PROJECT

Counterfeit products are low-quality imitations of genuine products can be difficult to identify, and they can cause serious harm to consumers who unknowingly purchase them. A fully functional and secure blockchain technology is suggested to detect counterfeit products is a promising approach to combat the global problem of facing new challenges like copying and counterfeiting that could harm the brand name, income, and consumer happiness of the firm.

By leveraging the immutability and transparency of blockchain, it becomes possible to create a tamper-proof record of every transaction that a product goes through without the need of any central authority. This record can be used to track the product's ownership and verify its authenticity, thereby reducing the risk of counterfeit products. By this business may stop worrying about fake goods by investing very little time and money. When consumers mistakenly believe that the product is a legitimate offering from the maker, manufacturers suffer severe damage to their brand value and reputation as a result of counterfeit items. To implement this system, every product is assigned a unique QR code that maintains information about the product's manufacturer, distributor, and retailer. Whenever a product changes ownership, the transaction is recorded on the blockchain, and the new owner's identity is verified using their public key. The blockchain also maintains the complete history of ownership of the product, making it easier to trace the origin of the product.

Registered user of this application can scan the QR code of the product to check the authenticity of it. The information displayed includes the history of ownership of that product which increases the customer confidence and does not effects the reputation of the brand. If the product is counterfeit, the customer will be alerted immediately. With the use of blockchain technology, it becomes difficult for counterfeiters to duplicate products and pass them off as original. Overall, it's an effective way to protect the integrity of the supply chain.

Blockchain does not allow any user to update the existing data each time blockchain will add data as a new block to existing data. So, it is impossible to delete or modify the data in the blockchain which leads to the security and protection of data. Blockchain helps to solve the problem of counterfeiting products.

1.1.1 PROPOSED SYSTEM

In response to the widespread increase in counterfeit products worldwide, it is necessary to develop a comprehensive application system to identify and reduce the unauthorized usage of them. The proposed system described in this paper involves storing the supply chain and ownership history of each product. This allows customers to access complete information about the product and determine its authenticity before making a purchase. To verify the products, QR codes will be used to add and confirm product information and to make payment when it is shipped to one of the authorized users. In addition, to ensure the integrity of the stored data, blockchain technology will be utilized to prevent any unauthorized changes. Thus, the proposed system employs blockchain and QR codes to detect counterfeit products.

1.2 FEATURES

- Identifying the real and fake product.
- Adding a product to blockchain.
- Shipping the product and paying ethers to it.

1.3 SOFTWARE REQUIREMENTS

- Language: Solidity, CSS, React JS, Web3 JS
- Code Editor: Visual Studio Code
- Operating System: Windows 8 or above
- Wallet: MetaMask
- Framework: Truffle
- Local Blockchain: Ganache

1.4 HARDWARE REQUIREMENTS

- Hard disk – 256 GB or above.
- Ram required – 4 GB or above
- Processor – i3 or above

CHAPTER 2
LITERATURE SURVEY

A. Akhtar, et al. proposes a blockchain-based solution for preventing product counterfeiting in the supply chain. The authors develop a system that uses blockchain to store product information and a smart contract to verify the authenticity of products.

A. Bayraktaroglu and M. O. Yıldız, et. al. provides an overview of current developments in fake product detection using blockchain technology. The authors review the literature on blockchain-based anti-counterfeiting systems and provide insights into the strengths and weaknesses of existing approaches.

Aini, Qurotul, et al. provides us the information about the QR codes, their texture, and authentication. The popular use of high-quality printing and scanning QR codes makes it easier to counterfeit important printed matter, such as important documents, the anti-counterfeit label on merchandise, packaging, etc.

Ali, Omar, et al. presents a comprehensive summary of blockchain technology, offering a detailed account of its evolution, architecture, and security.

Al-Farsi, Sana, Muhammad Mazhar Rathore, and Spiros Bakiras, et al. describes how blockchain is used in supply chain management to enhance product traceability, authenticity, and legality at a lower cost. By creating an immutable, accessible, and verifiable record of product movement throughout the supply chain, blockchain technology improves overall supply chain management.

Bhutta, Muhammad Nasir Mumtaz, et al. presents a comprehensive summary of blockchain technology, offering a detailed account of its evolution, architecture, and security.

Daniel M. Hall and Hunhevicz, Jens J., et. al. provides us with a simple flowchart which is useful to know that even if we need blockchain in our projects or not. It provides different scenarios and by bypassing these cases you come to know whether blockchain is needed or not.

Dursun, Taner, et al. describes how blockchain is used in supply chain management to enhance product traceability, authenticity, and legality at a lower cost. By creating an immutable, accessible,

and verifiable record of product movement throughout the supply chain, blockchain technology improves overall supply chain management.

F. M. Castro-Mejía, et al. provides a literature review of the use of blockchain technology for supply chain traceability. The authors review the existing literature on blockchain-based anti-counterfeiting systems and provide insights into the strengths and weaknesses of existing approaches.

G. K. Mandal, et al. presents a blockchain-based system for securing the food supply chain. The authors propose a system that uses blockchain to create a tamper-proof record of food products' movement from farm to table, making it difficult for counterfeiters to introduce fake products into the supply chain.

J. J. Rodriguez-Molina, et al. explores the application of blockchain technology in the pharmaceutical sector as a means of preventing the infiltration of counterfeit medications into the supply chain. Authors of [9] advocate for a system that employs blockchain to monitor the distribution of drugs from the manufacturer to the end-user, streamlining the identification and removal of fraudulent drugs from the supply chain.

J. Zhang, et al. proposes a system for detecting fake products using blockchain technology and deep learning. The authors develop a blockchain-based database that stores product information and a deep learning model that can detect counterfeit products based on product images.

Jambhulkar, Swaroop, et al. proposes a system that involves creating a unique QR code containing all relevant product information and storing it in a blockchain database. When a customer or distributor purchases the product, they can scan the QR code embedded in it. If the scanned code matches the stored QR code, the system confirms the product's authenticity. Conversely, if the scanned code does not match the stored QR code, the system identifies the product as counterfeit.

Rajendran et al. provides a thorough explanation of the various ways in which blockchain technology can be employed to detect counterfeit goods, exploring a range of blockchain-based systems and assessing their respective benefits and drawbacks for detecting fraudulent products. This is introduced in the paper [6].

Shreekumar, T., et al. proposes a system that involves embedding a QR code in a product during manufacturing, and allowing users to verify its authenticity by comparing the product's QR codes with the stored QR code.

Turjo, Manoshi Das, et al. describes how blockchain is used in supply chain management to enhance product traceability, authenticity, and legality at a lower cost. By creating an immutable, accessible, and verifiable record of product movement throughout the supply chain, blockchain technology improves overall supply chain management.

Wang et al. proposes a strategy for detecting counterfeit products in e-commerce through the use of blockchain technology. Specifically, the approach [7] relies on a reputation system that is built upon blockchain to identify and prevent the sale of fraudulent products on e-commerce platforms.

Wang et al. proposes a new method for implementing a traceability system [8] to combat counterfeiting in the food industry, utilizing a combination of blockchain, internet of things (IoT), and machine learning (ML) technologies to guarantee the legitimacy of food products.

CHAPTER 3

SYSTEM ANALYSIS

3.1 FEASIBILITY STUDY

The feasibility study of this project has revealed the project as follows: -

ECONOMIC FEASIBILITY

The project has shown the economic feasibility by the study of the fact that by the usage of blockchain technology which makes use of cloud platforms to store the chain of data in database, this reduces the cost for hardware, saves money. This project only requires a MySQL database in XAMPP server to store the authentication of the web application, it stores only the data of users who registered in this web application. Hence this project is economically feasible.

OPERATIONAL FEASIBILITY

Interface of this project be simple so that everyone should be able to use it in a user-friendly manner. The solution for the project that is suggested is acceptable. The operations involved in this project such as shipment and payment are acceptable and are satisfied by the organization. This shows the behavioral feasibility of the project.

TECHNICAL FEASIBILITY

Technical feasibility is carried out to determine whether the project is feasible in terms of software, hardware, personnel, and expertise, to handle the completion of the project. It considering resources for the proposed system as it does not require much hardware resources and the requested technology are possible to embed with the current technology(blockchain) and to is stable. The technology of the latest hence the system is also technically feasible.

TIME FEASIBILITY

The proposed project can be implemented fully within stipulated time frame. The project was decided to be done in three months and was thought to be feasible enough.

3.2 FUNCTIONAL REQUIREMENTS

- **Add a product:** Manufacturer adds a new product by generating a QR code, it provides all the details of the product.

- **Scan QR code:** Any authorized user can scan the QR to get the details of a product.

- **Shipment:** A manufacturer can ship a product to distributor then a block is added corresponding to this transaction, same in this way a new block is added to the chain with details of the current owner when a product is ship to retailer and end user from distributor and retailer respectively.

- **Payment:** When a product is shipped to a user, then he/she can able to ship that product to other user only if the payment is done.

3.3 NON-FUNCTIONAL REQUIREMENTS

3.3.1. Performance Requirements

Every transaction in the blockchain takes less than 10 seconds to add a node in the distributed ledger. This system is user friendly to make any operation in this application and it is interactive.

3.3.2. Safety Requirements

As smart contracts are developed using Blockchain Technology, the data is stored at multiple nodes. Since data is maintained at multiple locations, it will be safe during physical disasters also.

3.3.3. Security Requirements

To specify objectives and expectations to protect the service and data at the core of application. This allows information access to all designated nodes or members who can record, share, and view encrypted transactional data on their blockchain.

3.3.4. Software Quality Attributes

The application is flexible when it can run smoothly on any device, platform, or operating system. It is scalable as it uses vertical scalability for storing the ledger. This is flexible as it can easily adapt to future changes. As blockchain has decentralized and distributed network which leads to no single point of failure.

3.3.5. Flexibility

System is working easily on the Intranet by any user. The system can also work on other kind of technology with the little modification.

3.3.6. Efficiency

System is efficient enough to meet all kinds of requirements as required by the users. The system should not hang or lose its efficiency in any kind of worse conditions. It provides the correct output in all manners.

3.3.7. User Friendliness

System is user friendly, so that any user can use and access the system with easiness.

3.3.8. Availability

The system can operate 24 hours per week and 365 days a year. As long as the server is running. All the information will be in the database. Even though, the desktop is shut off information still exist in the database.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION OF SYSTEM DESIGN

Systems design is an interdisciplinary engineering activity that enables the realization of successful systems. This system may be denned as an integrated set of components that accomplish a defined objective. The process of systems design includes defining software and hardware architecture, modules, interfaces, and data to enable a system to satisfy a set of well-specified operational requirements.

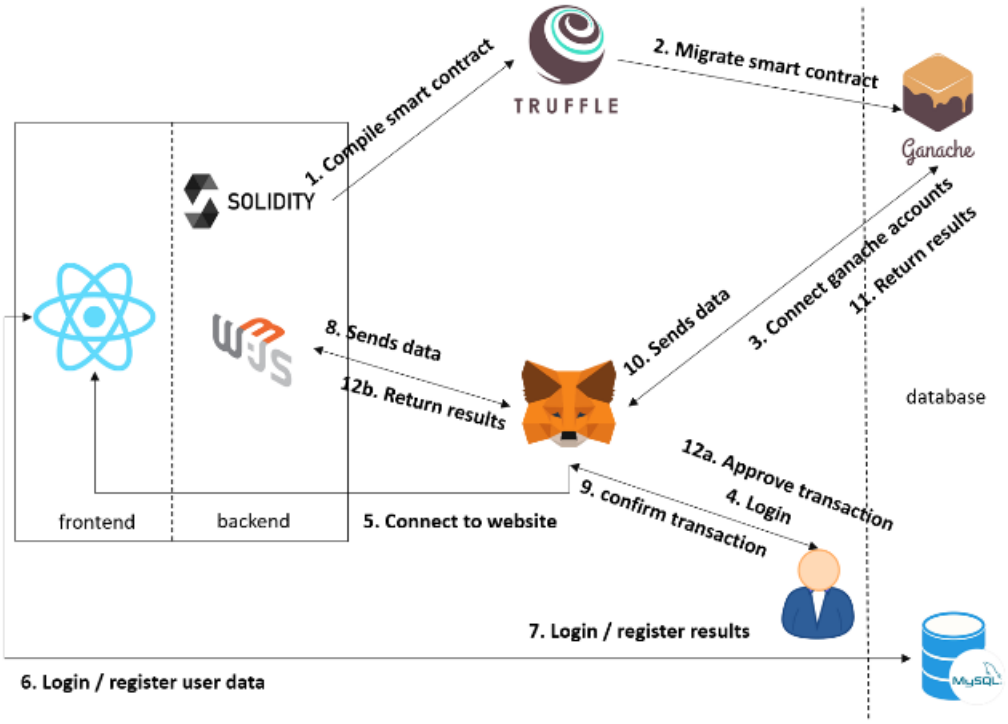


Figure 4.1: System Architecture

4.2 USE CASE DIAGRAM

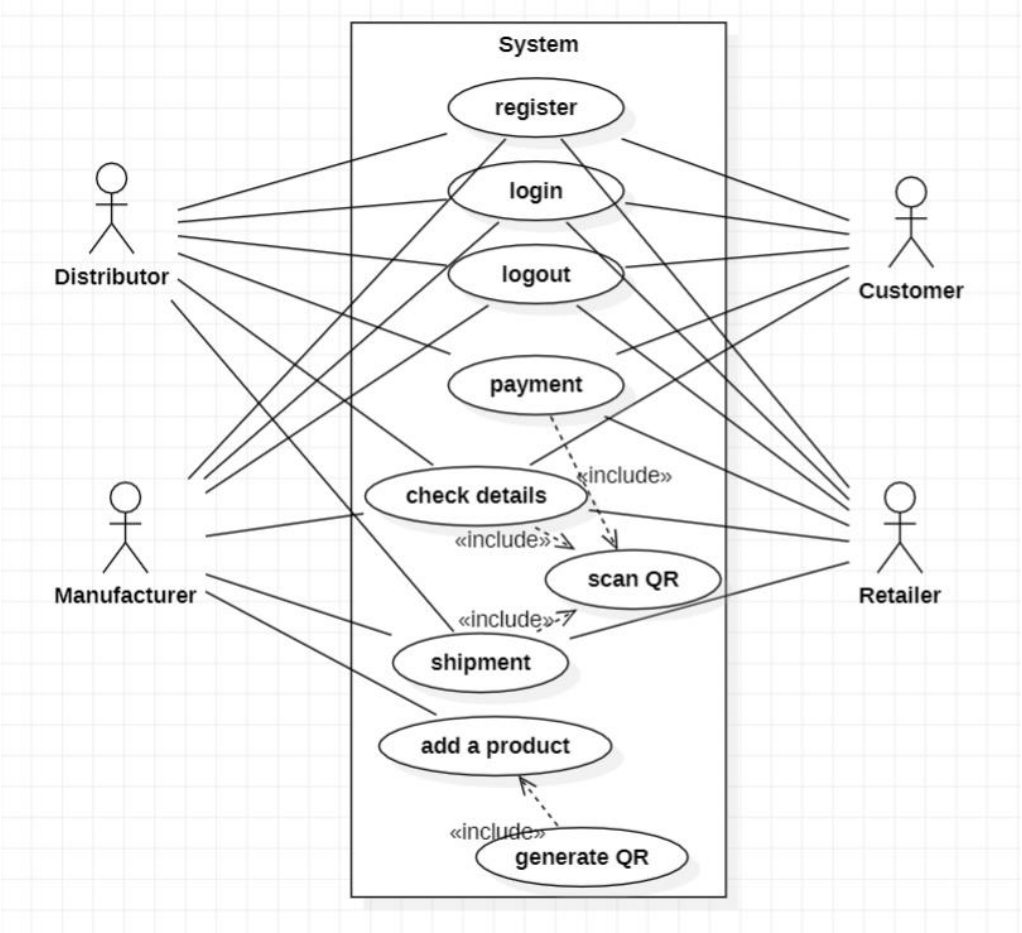


Figure 4.2: Use Case Diagram

4.3 ACTIVITY DIAGRAM

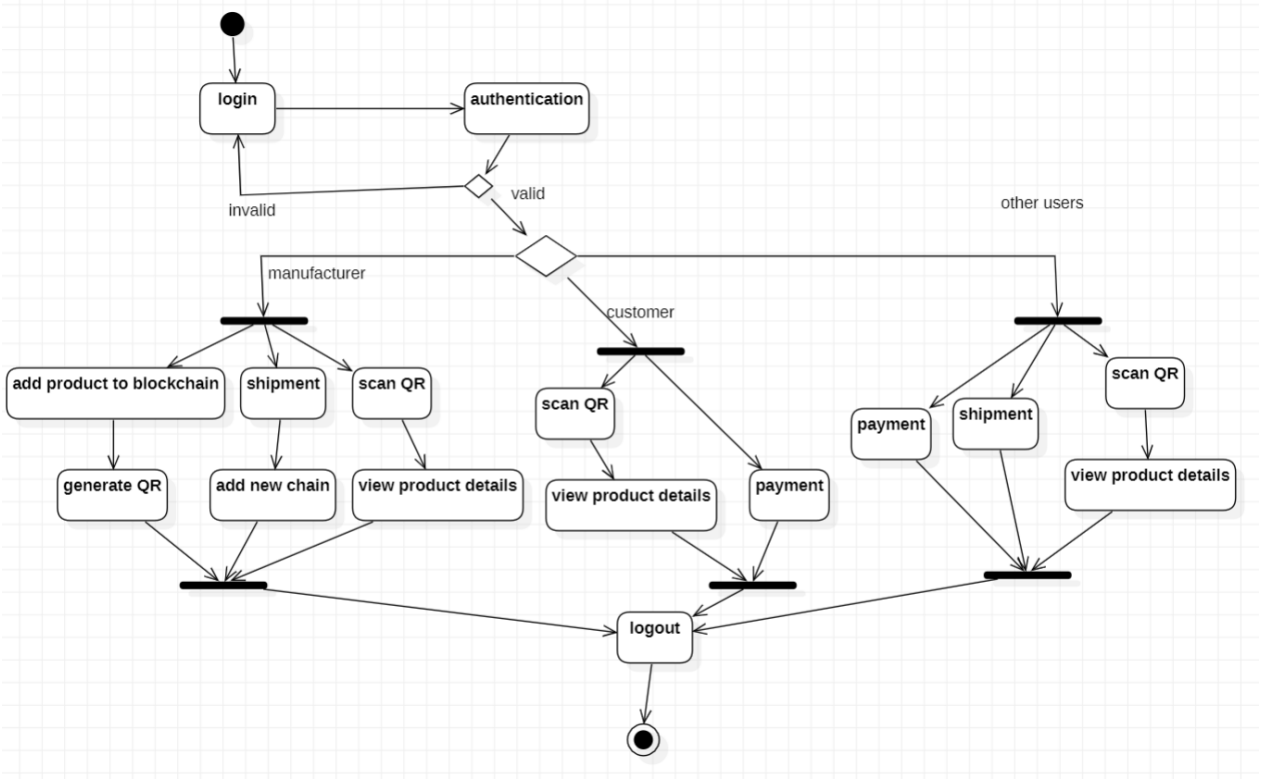


Figure 4.3: Activity Diagram

4.5 SEQUENCE DIAGRAM: MANUFACTURER

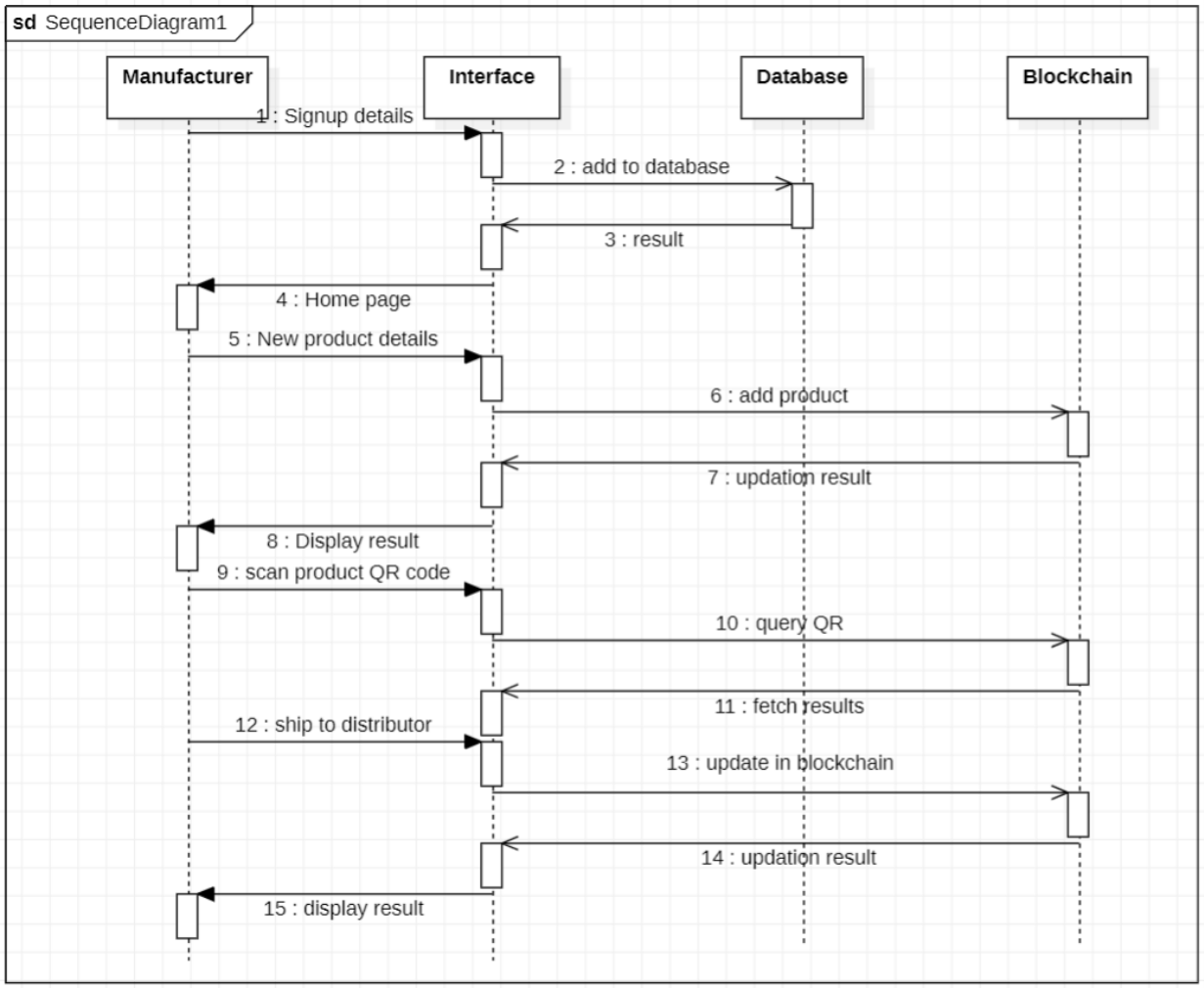


Figure 4.4: Sequence Diagram of Manufacturer

4.6 SEQUENCE DIAGRAM: CUSTOMER

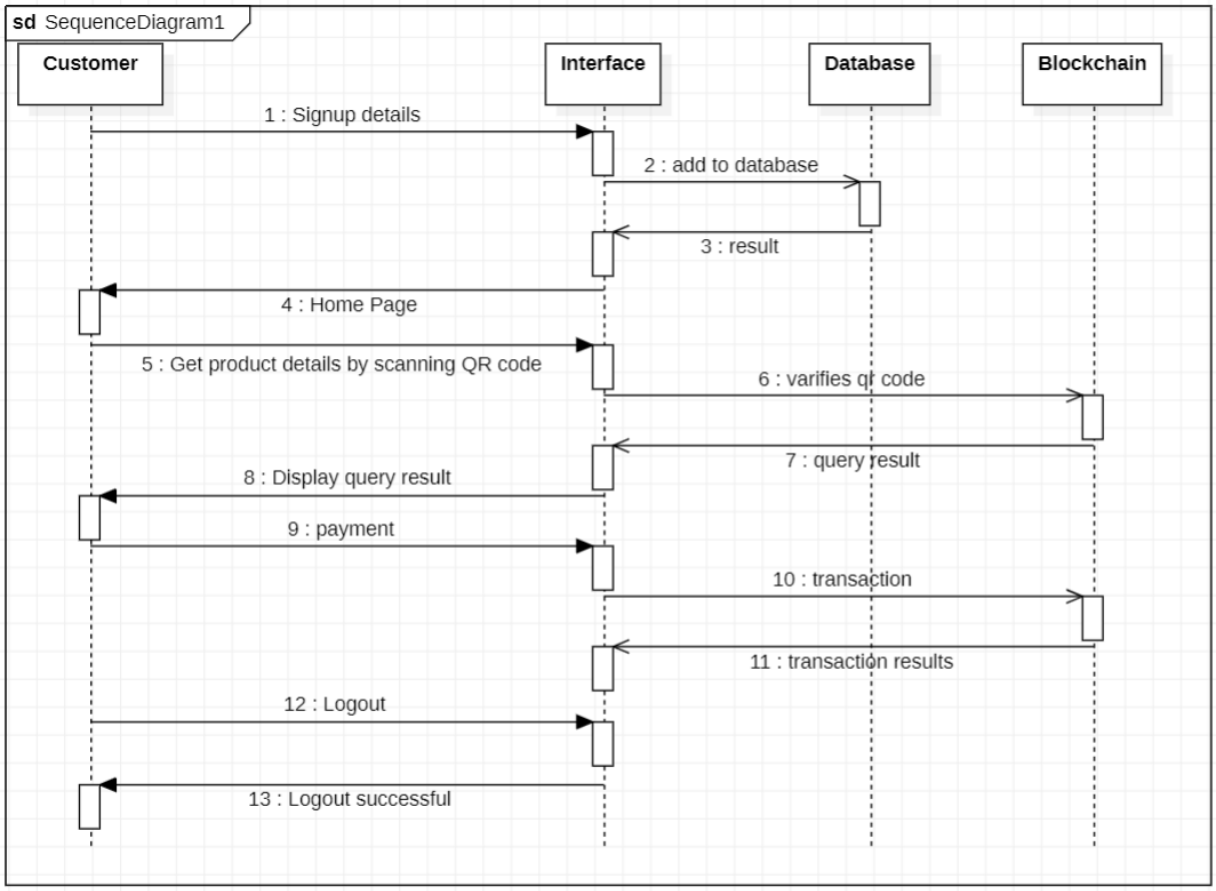


Figure 4.5: Sequence Diagram of Customer

4.7 SEQUENCE DIAGRAM: DISTRIBUTOR / RETAILER

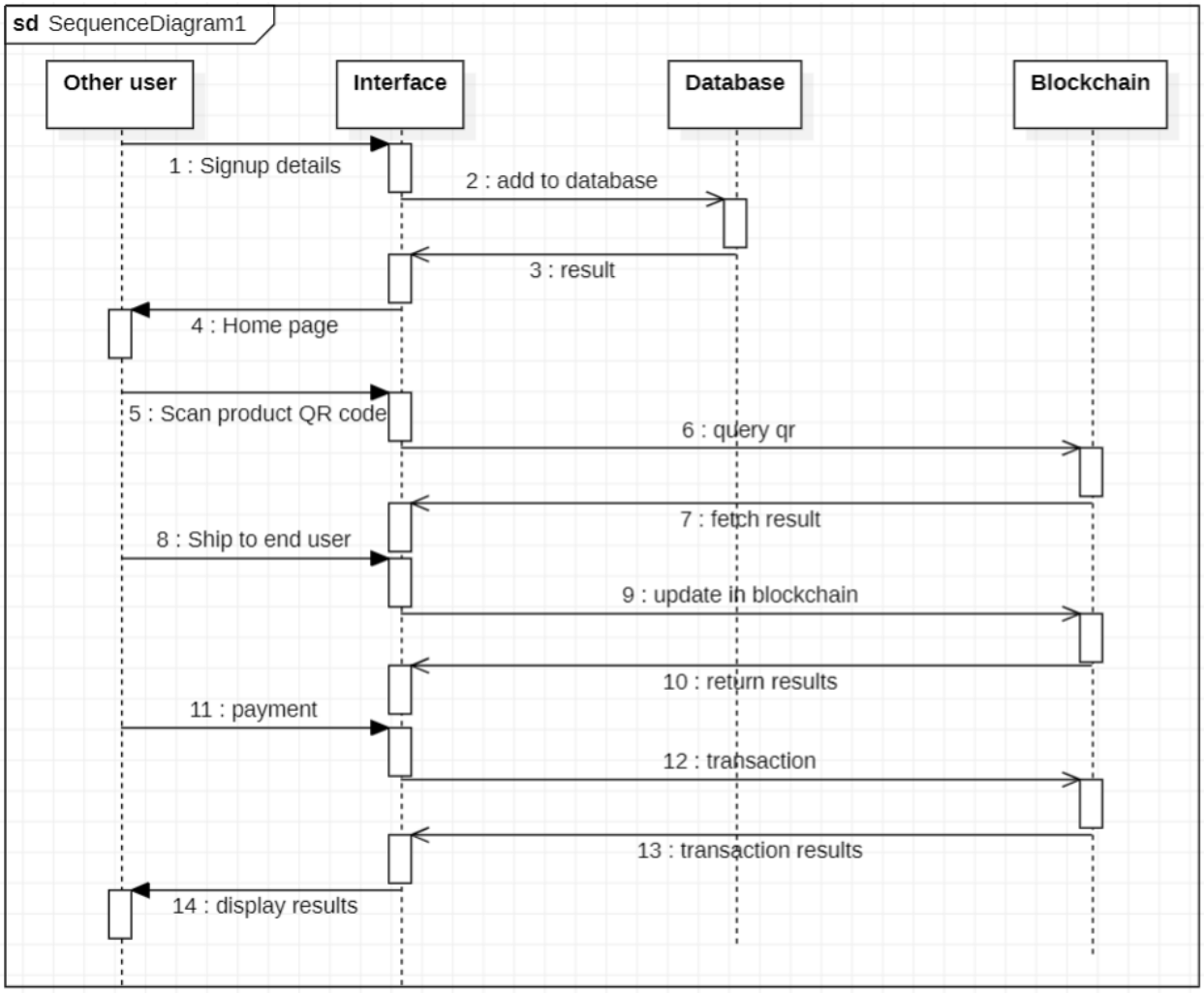


Figure 4.6: Sequence Diagram of Distributor/Retailer

4.8 CLASS DIAGRAM

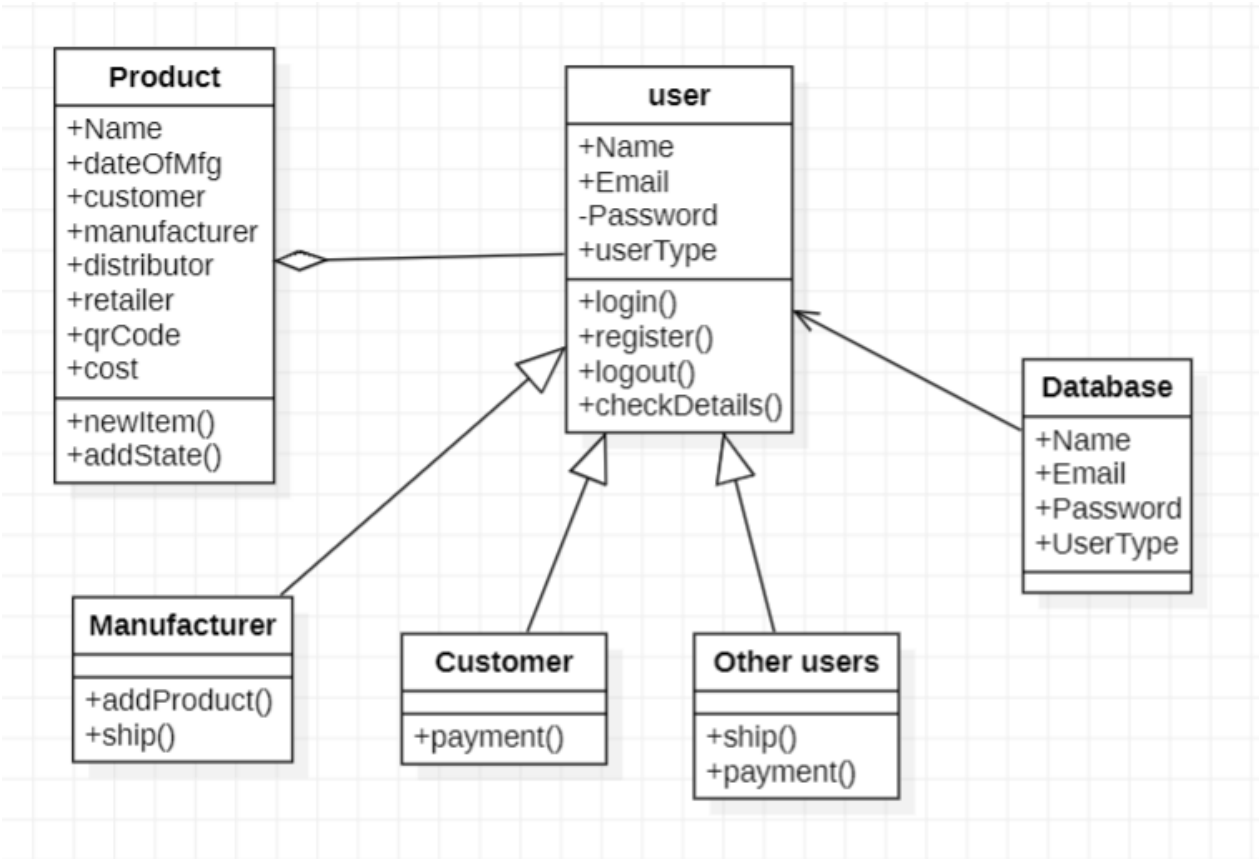


Figure 4.7: Class Diagram

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 TECHNOLOGIES USED

- Blockchain
- MetaMask
- Ganache
- React JS
- Web3.js
- Solidity

5.2 BLOCKCHAIN

A blockchain is a distributed database or ledger that is shared among the nodes of a computer network. As a database, a blockchain stores information electronically in digital format. Blockchains are best known for their crucial role in cryptocurrency systems, such as Bitcoin, for maintaining a secure and decentralized record of transactions. One key difference between a typical database and a blockchain is how the data is structured. A blockchain collects information together in groups, known as blocks, that hold sets of information. Blocks have certain storage capacities and, when filled, are closed and linked to the previously filled block, forming a chain of data known as the blockchain. All new information that follows that freshly added block is compiled into a newly formed block that will then also be added to the chain once filled. The goal of blockchain is to allow digital information to be recorded and distributed, but not edited.

Blockchain technology achieves decentralized security and trust in several ways. To begin with, new blocks are always stored linearly and chronologically. That is, they are always added to the "end" of the blockchain. After a block has been added to the end of the blockchain, it is extremely difficult to go back and alter the contents of the block unless a majority of the network has reached a consensus to do so. That's because each block contains its own hash, along with the hash of the block before it, as well as the previously mentioned timestamp. Hash codes are created by a mathematical function that turns digital information into a string of numbers and letters. If that information is edited in any way, then the hash code changes as well.

Succeeding with such a hack would require that the hacker simultaneously control and alter 51% or more of the copies of the blockchain so that their new copy becomes the majority copy and, thus, the agreed-upon chain. Such an attack would also require an immense amount of money and resources, as they would need to redo all of the blocks because they would now have different timestamps and hash codes.

The four key concepts behind blockchain are:

- Shared ledger: A shared ledger is an "append-only" distributed system of record shared across a business network. "With a shared ledger, transactions are recorded only once, eliminating the duplication of effort that's typical of traditional business networks."
- Permissions: Permissions ensure that transactions are secure, authenticated, and verifiable. "With the ability to constrain network participation, organizations can more easily comply with data protection regulations, such as those stipulated in the Health Insurance Portability and Accountability Act (HIPAA)" and the EU General Data Protection Regulation (GDPR).
- Smart contracts: A smart contract is "an agreement or set of rules that govern a business transaction; it's stored on the blockchain and is executed automatically as part of a transaction."
- Consensus: Through consensus, all parties agree to the network-verified transaction. Blockchains have various consensus mechanisms, including proof of stake, multi signature, and PBFT (Practical Byzantine Fault Tolerance).

Each blockchain network has various participants who play these roles, among others:

- Blockchain users: Participants (typically business users) with permissions to join the blockchain network and conduct transactions with other network participants.
- Regulators: Blockchain users with special permissions to oversee the transactions happening within the network.
- Blockchain network operators: Individuals who have special permissions and authority to define, create, manage, and monitor the blockchain network.
- Certificate authorities: Individuals who issue and manage the different types of certificates required to run a permissioned blockchain.

Underlying blockchain mechanisms are complex. Below are the steps involved in the mechanism of blockchain to add a block in the ledger.

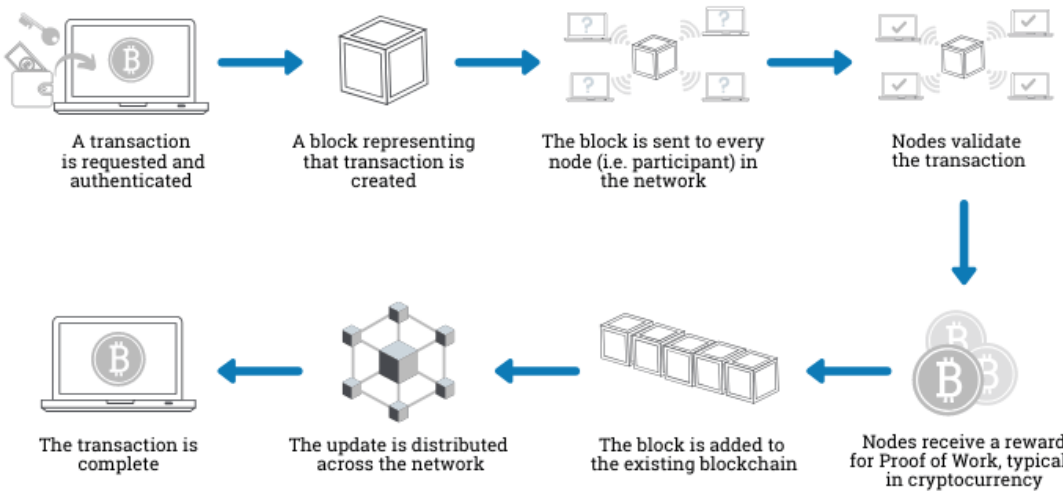


Figure 5.2.1. Working of Blockchain

Step 1 – Record the transaction

A blockchain transaction shows the movement of physical or digital assets from one party to another in the blockchain network. It is recorded as a data block and can include details like these:

- Who was involved in the transaction?
- What happened during the transaction?
- When did the transaction occur?
- Where did the transaction occur?
- Why did the transaction occur?
- How much of the asset was exchanged?
- How many pre-conditions were met during the transaction?

Step 2 – Gain consensus

Most participants on the distributed blockchain network must agree that the recorded transaction is valid. Depending on the type of network, rules of agreement can vary but are typically established at the start of the network.

Step 3 – Link the blocks

Once the participants have reached a consensus, transactions on the blockchain are written into blocks equivalent to the pages of a ledger book. Along with the transactions, a cryptographic hash is also appended to the new block. The hash acts as a chain that links the blocks together. If the contents of the block are intentionally or unintentionally modified, the hash value changes, providing a way to detect data tampering.

Thus, the blocks and chains link securely, and you cannot edit them. Each additional block strengthens the verification of the previous block and therefore the entire blockchain. This is like stacking wooden blocks to make a tower. You can only stack blocks on top, and if you remove a block from the middle of the tower, the whole tower breaks.

Step 4 – Share the ledger

The system distributes the latest copy of the central ledger to all participants.

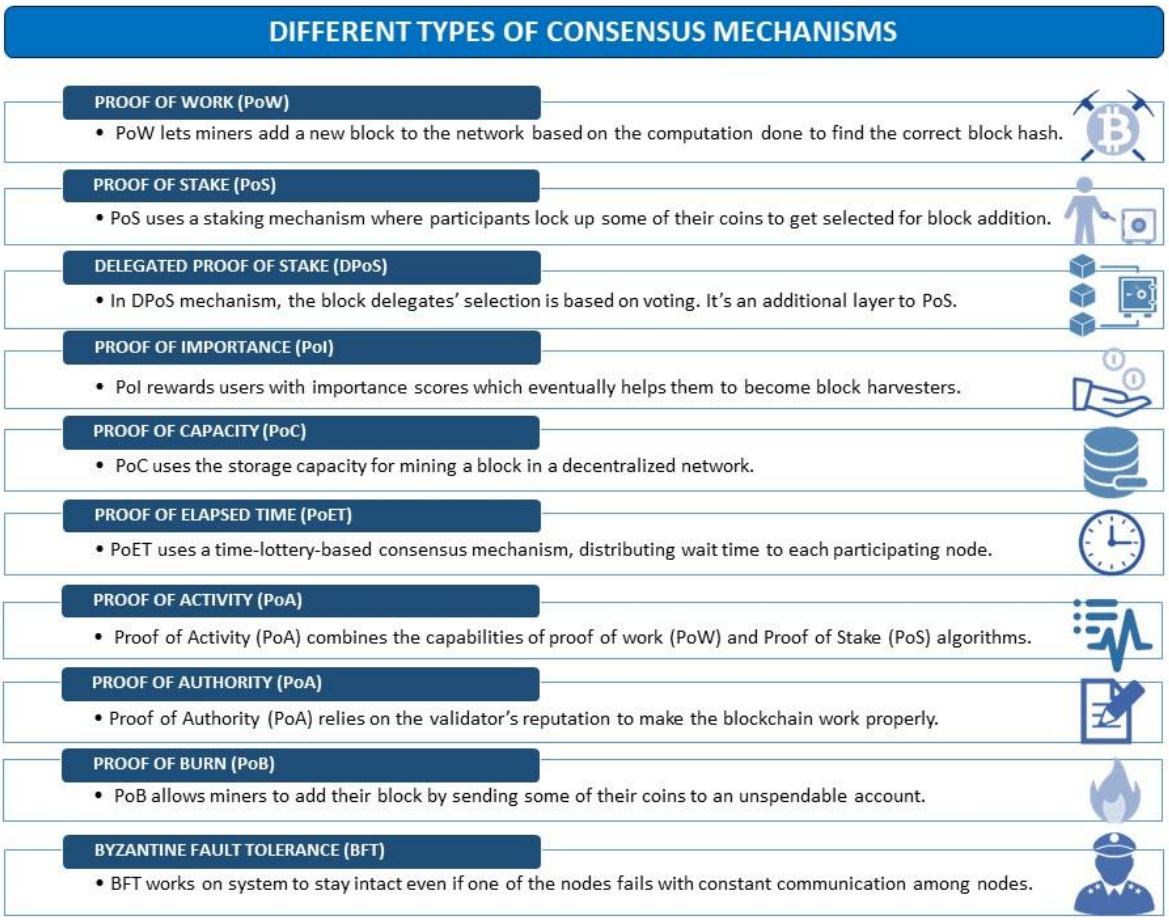


Figure 5.2.2. Different Consensus Mechanisms

There are four types of blockchains:

1. Public Blockchains: Public blockchains are open, decentralized networks of computers accessible to anyone wanting to request or validate a transaction (check for accuracy). Those (miners) who validate transactions receive rewards. Public blockchains use proof-of-work or proof-of-stake consensus mechanisms (discussed later). Two common examples of public blockchains include the Bitcoin and Ethereum (ETH) blockchains.

2. Private Blockchains: Private blockchains are not open, they have access restrictions. People who want to join require permission from the system administrator. They are typically governed by one entity, meaning they're centralized. For example, Hyperledger is a private, permissioned blockchain.
3. Hybrid Blockchains or Consortiums: Consortiums are a combination of public and private blockchains and contain centralized and decentralized features. For example, Energy Web Foundation, Dragonchain, and R3.
Take note: There isn't a 100 percent consensus on whether these are different terms. Some make a distinction between the two, while others consider them the same thing.
4. Sidechains: A sidechain is a blockchain running parallel to the main chain. It allows users to move digital assets between two different blockchains and improves scalability and efficiency. An example of a sidechain is the Liquid Network.

5.3 METAMASK

MetaMask is a software cryptocurrency wallet used to interact with the Ethereum blockchain. It allows users to access their Ethereum wallet through a browser extension or mobile app, which can then be used to interact with decentralized applications.

MetaMask also lets the user create and manage their own identities (via private keys, local client wallet and hardware wallets like Trezor™), so when a DApp wants to perform a transaction and write to the blockchain, the user gets a secure interface to review the transaction, before approving or rejecting it.

Websites or other decentralized applications are able to connect, authenticate, and/or integrate other smart contract functionality with a user's MetaMask wallet (and any other similar blockchain wallet browser extensions) via JavaScript code that allows the website to send action prompts, signature requests, or transaction requests to the user through MetaMask as an intermediary. MetaMask also helps warn you when you navigate to sites that are known to have engaged in phishing, or that have names that are suspiciously similar to popular phishing targets.

Below are the steps to connect Ganache to MetaMask wallet:

1. Click Network select the option and choose Custom RPC.
2. You will be asked for the below details:
    - Network Name: This is your Network and name it as anything relevant to your

project.

- New RPC URL: This can be obtained from your Server tab of the Ganache Settings section. It has to be in the format http://<hostname>:<port>. Since we are running on local, it would mostly be http://127.0.0.1:7545
- Chain ID: For some reason, the default chain id for Ganache in MetaMask is 1337.
- Currency Symbol: This is optional and no harm in choosing one. I will leave it blank for now.
- Block Explorer URL: At this time, the GitHub issue to make it possible is being worked upon. I will leave it blank for now.

Now, pick up one of the accounts from Ganache workspace and add import it to MetaMask wallet, so that you can use the available ETH using MetaMask wallet. Follow the below steps:

1. Open MetaMask wallet and Click on the Account Circle.
2. Below the account list, choose Import Account.
3. Pick one address from Ganache Desktop UI and copy the Private key from the Key symbol at the right end.
4. Paste private key string to MetaMask.

5.4 GANACHE

Ganache is actually a component of the Truffle Suite framework along with the other components, Truffle and Drizzle. The functionality of Ganache as a high-end development tool, used for running your personal local blockchain network for developing decentralized applications on Ethereum as well as Corda. Ganache serves a vital role in all stages of the development process with many plausible advantages. The biggest advantages of using Ganache smart contract development would refer to the facility for developing, testing, and deploying your smart contracts and DApp projects in a deterministic and safe environment.

Ganache is a private Ethereum blockchain environment that allows to you emulate the Ethereum blockchain so that you can interact with smart contracts in your own private blockchain. Here are some features that Ganache provides:

- Displays blockchain log output.
- Provides advanced mining control.
- Built-in block explorer.

- Ethereum blockchain environment.
- Ganache has a desktop application as well as a command-line tool.

Following are the steps to connect truffle with ganache:

1. Create a new workspace in Ganache and add truffle-config file of the project into it.
2. Then start the workspace, it is now ready to get all transaction blocks and hash values of blocks into it and can have the log file and migrated contracts.
3. Create the artifact file of the smart contract that used to deploy the contract.
4. Open the terminal in VS Code and move to the directory where truffle is installed.
5. First compile the contract file using the command “truffle compile <contract_name>”. It creates a .json file of that contract.
6. After successful compilation, migrate the contracts to Ganache workspace using the command “truffle migrate” when it is the first migration of contract, otherwise to update the migration in Ganache use the command “truffle migrate --reset”.
7. Successful migration of smart contracts into Ganache creates a first contract call block, now we can start our transactions that generates new blocks with the block number.

5.5 REACT JS

ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front-end library which is responsible only for the view layer of the application. The main objective of ReactJS is to develop User Interfaces (UI) that improves the speed of the apps. It uses virtual DOM (JavaScript object), which improves the performance of the app. The JavaScript virtual DOM is faster than the regular DOM. We can use ReactJS on the client and server-side as well as with other frameworks. It uses component and data patterns that improve readability and helps to maintain larger apps.

The important features of ReactJS are as following.

- JSX
- Components
- One-way Data Binding
- Virtual DOM
- Simplicity
- Performance

JSX: JSX stands for JavaScript XML. It is a JavaScript syntax extension. Its an XML or HTML like syntax used by ReactJS. This syntax is processed into JavaScript calls of React Framework. It extends the ES6 so that HTML like text can co-exist with JavaScript react code. It is not necessary to use JSX, but it is recommended to use in ReactJS.

Components: ReactJS is all about components. ReactJS application is made up of multiple components, and each component has its own logic and controls. These components can be reusable which help you to maintain the code when working on larger scale projects.

One-way Data Binding: ReactJS is designed in such a manner that follows unidirectional data flow or one-way data binding. The benefits of one-way data binding give you better control throughout the application. If the data flow is in another direction, then it requires additional features. It is because components are supposed to be immutable and the data within them cannot be changed. Flux is a pattern that helps to keep your data unidirectional. This makes the application more flexible that leads to increase efficiency.

Virtual DOM: A virtual DOM object is a representation of the original DOM object. It works like a one-way data binding. Whenever any modifications happen in the web application, the entire UI is re-rendered in virtual DOM representation. Then it checks the difference between the previous DOM representation and new DOM. Once it has done, the real DOM will update only the things that have actually changed. This makes the application faster, and there is no wastage of memory.

Simplicity: ReactJS uses JSX file which makes the application simple and to code as well as understand. We know that ReactJS is a component-based approach which makes the code reusable as your need. This makes it simple to use and learn.

Performance: ReactJS is known to be a great performer. This feature makes it much better than other frameworks out there today. The reason behind this is that it manages a virtual DOM. The DOM is a cross-platform and programming API which deals with HTML, XML or XHTML. The DOM exists entirely in memory. Due to this, when we create a component, we did not write directly to the DOM. Instead, we are writing virtual components that will turn into the DOM leading to smoother and faster performance.

Truffle box named “react” box is used to install combination of both truffle and create-react-dom as truffle and client end. This box also provides a sample SimpleStorageContract file that connects

with MetaMask, Ganache and the corresponding front-end react app which gives more feasibility to work with blockchain using React JS.

5.6 WEB3.JS

Web3.js is a collection of libraries that allow you to interact with a local or remote Ethereum node using HTTP, IPC or WebSocket. This can be actions like sending Ether from one user to another, checking data from smart contracts, creating smart contracts, among other things. Ethereum nodes provide interfaces to users in order to complete transactions: of which, nodes receive this information through a JSON RPC interface. This is an encoding format that allows running processes to receive new and verify existing data. Web3.js helps to make the process of running and selecting nodes participating in the Ethereum network simpler and easier to grasp.

This can be actions like sending Ether from one user to another, checking data from smart contracts, creating smart contracts, among other things. Ethereum nodes provide interfaces to users in order to complete transactions: of which, nodes receive this information through a JSON RPC interface. This is an encoding format that allows running processes to receive new and verify existing data. Web3.js helps to make the process of running and selecting nodes participating in the Ethereum network simpler and easier to grasp.

Generally, those creating DApp or integrated web browser applications into the Ethereum blockchain utilize the MetaMask browser extension in conjunction with Web3.js. MetaMask is an Ethereum wallet that is hosted in-browser and natively puts a Web3 provider object into said browser: which, in brief, a Web3 provider object is a data-structure that provides links to publicly accessible Ethereum nodes. Using Web3.js and MetaMask, users and developers are able to manage private keys and verify transactions directly from their preferred browser.

5.7 SOLIDITY

Solidity is an object-oriented, high-level language for implementing smart contracts. Smart contracts are programs which govern the behavior of accounts within the Ethereum state. Solidity is a curly-bracket language designed to target the Ethereum Virtual Machine (EVM). It is influenced by C++, Python and JavaScript. Solidity is statically typed, supports inheritance, libraries and complex user-defined types among other features.

With Solidity you can create contracts for uses such as voting, crowdfunding, blind auctions, and

multi-signature wallets. When deploying contracts, you should use the latest released version of Solidity. Apart from exceptional cases, only the latest version receives security fixes. Furthermore, breaking changes as well as new features are introduced regularly. We currently use a 0.y.z version number to indicate this fast pace of change.

Ethereum is a decentralized i.e., blockchain platform that runs smart contracts i.e., applications that run exactly as programmed without any possibility of downtime, censorship, fraud or third-party interference. A smart contract is a computer protocol intended to digitally facilitate, verify, or enforce the negotiation or performance of a contract. Smart contracts allow the performance of credible transactions without third parties. These transactions are trackable and irreversible.

The Ethereum Virtual Machine, also known as EVM, is the runtime environment for smart contracts in Ethereum. The Ethereum Virtual Machine focuses on providing security and executing untrusted code by computers all over the world. The EVM specialized in preventing Denial-of-service attacks and ensures that programs do not have access to each other's state, ensuring communication can be established without any potential interference. The Ethereum Virtual Machine has been designed to serve as a runtime environment for smart contracts based on Ethereum.

5.8 SAMPLE CODE

FakePro.sol

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.4.22 <0.9.0;

contract FakePro {

   struct State{
      string owner;
      address person;
      string date;
      bool ispay;
   }

   struct Product{
      address creator;
      string productName;
      uint256 cost;
      uint256 productId;
      string date;
      uint256 totalStates;
   }

   mapping(string => Product) allProducts;
   mapping (uint256 => State) positions;
   uint256 items=1;

   function concat(string memory _a, string memory _b) public pure returns (string memory){
      return string(bytes.concat(bytes(_a), " ", bytes(_b)));
   }

   function newItem(string memory _text, uint256 _cost, string memory info, string memory _date) public
returns (uint256) {
      require(allProducts[_text].totalStates==0,"Already this product exists");
      Product memory newITem = Product({creator: msg.sender, totalStates: 1, productName: _text, cost:
_cost, productId: items, date: _date});
      allProducts[_text]=newITem;
      State memory newState = State({person: payable(msg.sender), owner: info, date: _date, ispay: true});
      positions[ allProducts[_text].totalStates-1 ]=newState;
      items = items+1;
      return (items-1);
   }

   function getPaymentStatus(string memory _text) public view returns(bool){
      return positions[allProducts[_text].totalStates-1].ispay;
   }

   function getPayDetais(string memory _text) public view returns(bool, address, uint256){
      return (
         positions[allProducts[_text].totalStates-1].ispay,
```

```
            positions[allProducts[_text].totalStates-2].person,
            allProducts[_text].cost
        );
    }

    function pay(string memory _text) public payable{
        positions[allProducts[_text].totalStates-1].ispay=true;
        payable(positions[allProducts[_text].totalStates-2].person).transfer(msg.value);
    }

    function getLatestOwner(string memory _text) public view returns (string memory){
        return positions[allProducts[_text].totalStates-1].owner;
    }

    function getTotalStates(string memory _text) public view returns (uint256){
        return allProducts[_text].totalStates ;
    }

    function addState(string memory _text, string memory info, address _person, string memory _date)
public returns (string memory) {
        require(positions[allProducts[_text].totalStates-1].person==msg.sender,"Ensure your account
address");
        State memory newState = State({person: payable(_person), owner: info, date: _date, ispay: false});
        positions[ allProducts[_text].totalStates ]=newState;
        allProducts[_text].totalStates = allProducts[_text].totalStates +1;
        return info;
    }

    function searchProduct(string memory _text) public view returns (string memory) {

        require(allProducts[_text].productId<items && allProducts[_text].productId>0,"No data available");
        string memory output="";

        for (uint256 j=0; j<allProducts[_text].totalStates; j++){
            output=concat(output, positions[j].owner);
            output=concat(output, ":");
            output=concat(output, positions[j].date);
            output=concat(output, "::");
        }
        return output;
    }

}
```

FakePro.json

Add.jsx

```
import QRCode from "qrcode";
import {useState} from "react";
import useEth from "../contexts/EthContext/useEth";
import "./Add.css";

const Add = ({setAddOpen, username}) => {

   const [text, setText] = useState('');
   const [cost,setCost] = useState('');
   const [imageUrl, setImageUrl] = useState('');
   const { state: { contract, accounts } } = useEth();
   const generateQrCode = async (e) => {
     const name = sessionStorage.getItem("name");
     console.log(name);
     var today = new Date();
     console.log(cost);
     const date = today.getFullYear() + '-' + (today.getMonth() + 1) + '-' + today.getDate() +' ' +
today.getHours() + ':' + today.getMinutes() +':'+ today.getSeconds();
     if(sessionStorage.getItem("addr")==accounts[0]){
     console.log(date);
       try {
         await contract.methods.newItem(text,cost,name,date).send({ from: accounts[0] }).then(async receipt
=> {
             console.log(receipt);
             const response = await QRCode.toDataURL(text);
             setImageUrl(response);
           });
       }catch (error) {
         alert("Product already exists");
         console.log(error.message);
       }
     }else{
       alert("Ensure your account address and proceed further");
     }
   }

   return (<div className="addBackground">
   <div className="addContainer">
   <div className="heading">Counterfeit Product Detection</div>
     <div className="add">Add a new product</div>
     <div className="body">
       <input type="text" placeholder="Enter Product Name" onChange={(e) => setText(e.target.value)}/>
       <input type="number" placeholder="Enter Product Cost "
onChange={(e)=>{setCost(e.target.value)}}/>
       <button className="generate" variant="contained" color="primary" onClick={() =>
generateQrCode()}>
         Generate</button>
       <br/>
       <br/>
```

```
      {imageUrl ? (<a href={imageUrl} download="qr.png">
                  <img src={imageUrl} alt="img"/>
                  </a>) : null}
      </div>
      <div className="footer">
        <button
          onClick={() => {
            setAddOpen(false);
          }}
          id="cancelBtn"
        >
          Cancel
        </button>

        <button onClick={()=>{setAddOpen(false);}} id="donebtn" disabled={!imageUrl}
>Done</button>

    </div></div></div>);
}
export default Add;
```

Ship.jsx

```
import { useState, useRef, useEffect } from "react";
import QrScanner from "qr-scanner";
import useEth from "../contexts/EthContext/useEth";
import Axios from "axios";
import "./Ship.css";

const Ship = ({ setShipOpen }) => {
   const [data, setData] = useState('');
   const [file,setFile] = useState(null);
   const [state,setState] =useState('');
   const [isState,setIsState] = useState(false);
   const [payer,setPayer] = useState('');
   const [addr, setAddr] = useState('');
   const qrRef = useRef(null);
   const type=sessionStorage.getItem("role");

   const { state: { contract, accounts }, web } = useEth();

   const handleClick =() => {
      qrRef.current.click();
   }

   const retrieveFile = async (e) => {
      const qrdata = e.target.files[0]; //files array of files object
      setFile(qrdata);
      await QrScanner.scanImage(qrdata).then(result => {
        setData(result);})
      .catch(error => console.log(error || 'No QR code found.'));

   };

   useEffect(()=>{
    data && isEligible();
   },[data]);

   const isEligible = async () => {
    const name = sessionStorage.getItem("name");
    console.log(name);
    console.log(web.eth);

    try{
    const eligible = await contract.methods.getLatestOwner(data).call({ from: accounts[0]});
    console.log(eligible);
    console.log(name);
    if(eligible==name){
      const isPay = await contract.methods.getPaymentStatus(data).call({from: accounts[0]});
      if(isPay){
      getRole();
      }else{
        alert("Complete your payment for the product "+data+" to proceed with shipment");
        setShipOpen(false);
```

```
   }
  }
  else{
   setState("Current owner of product "+ data +" is "+eligible);
  }
  }catch(error){
   setState("Fake Product!!!!!!!!!! ");
  }
}

const getRole = () =>{

  var role=1;
if(type==1){
  role=2;
}else if(type==2){
  role=3;
}else if(type==3){
  role=4;
}
Axios.post("http://localhost:3001/home", {
  role: role,}
).then(async (response)=>{
    const prodname=data;
    console.log(data);
        const states = await contract.methods.getTotalStates(prodname).call({ from: accounts[0]});
        console.log("role");
        console.log(type);
        console.log("states");
        console.log(states);

        if(type!=states){
            if(states==0){
                setState("No such product. It is Fake!!!!!!!!!!!");
            }else if(states==1){
                setState("Product at Manufacturer");
            }else if(states==2){
                setState("Product at Distributor");
            }else if(states==3){
                setState("Product at Retailer");
            }else{
                setState("Product at Customer");
            }
        }else{
            setIsState(true);
            let select = document.querySelector("#selectNumber");
    for(let i=0;i<response.data.length;i++){
        let opt = response.data[i].username;
        let val = response.data[i].address;
        let e1 = document.createElement("option");
        e1.textContent = opt;
        console.log(opt);
        e1.value = [val, opt];
```

```
          console.log(val);
          select.appendChild(e1);
        }
          }
         });
    }

    const handleSelect = (e) => {
      const l1=(e.target.value).split(",");
      setPayer(l1[1]);
      setAddr(l1[0]);
    }

    const handleSubmit = async () => {
       const name = sessionStorage.getItem("name");
      console.log(name);
      var today = new Date();
      const date = today.getFullYear() + '-' + (today.getMonth() + 1) + '-' + today.getDate() +' ' +
today.getHours() + ':' + today.getMinutes() +':'+ today.getSeconds();
      console.log(data);
      console.log(date);
      const user = payer;
      console.log(user);
      console.log(addr);
      try{
        if(sessionStorage.getItem("addr")==accounts[0]){
        const receipt = await contract.methods.addState(data, user, addr, date).send({ from: accounts[0] });
      console.log(receipt);
      alert("Successfully shipped to "+addr);}
      else{
        setState("ENsure your account address");
      }

      setShipOpen(false);
      }catch(error){
        console.log(error.message);
        console.log(error);
        setState("No such product. It is Fake!!!!!!!!!!");
      }
    }

    return <div className="shipBackground">
    <div className="shipContainer">
    <div className="heading">Counterfeit Product Detection</div>
    <div className="ship">Ship a Product</div>
      <div className="body">
          <button type="button" onClick={handleClick} hidden="hidden">Scan QR code</button>
        <input
          type="file"
          id="file-upload"
          name="data"
          className="d-none"
          accept=".png, .jpg, .jpeg"
```

```
          ref={qrRef}
          onChange={retrieveFile}
        /><br/>
        <form id="myForm">
        <select id="selectNumber" onChange={(e) => handleSelect(e)} required>
            <option className="address">--Ship to Person--</option>
          </select>
          { !isState && (<p>{state}</p>)}
         </form>
         <button onClick={handleSubmit} disabled={!data}>Ship</button>
        </div>
        <div className="footer">
          <button
            onClick={() => {
              setShipOpen(false);
            }}
            id="cancelBtn"
          >
            Cancel
          </button>
          <button onClick={()=>{setShipOpen(false);}} id="donebtn">Done</button>
        </div>
    </div></div>;

}

export default Ship;
```

Check.jsx

```
import { useState, useRef } from "react";
import QrScanner from "qr-scanner";
import useEth from "../contexts/EthContext/useEth";
import "./check.css";

const Check = ({setCheckOpen}) => {
   const [data, setData] = useState('');
   const [file,setFile] = useState(null);
   const [details,setDetails] = useState('');
   const qrRef = useRef(null);
   const { state: { contract, accounts } } = useEth();

   const handleSubmit = async () => {
    const prodname=data;
    console.log(prodname);
    try{
      const receipt = await contract.methods.searchProduct(prodname).call({ from: accounts[0] });
    console.log(receipt);
    setDetails(receipt);
    }catch(error){
      setDetails("No such product. It is Fake!!!!!!!!!!");
    }
   }

   const handleClick =() => {
    qrRef.current.click();
   }

   const retrieveFile = async (e) => {
    const qrdata = e.target.files[0]; //files array of files object
    setFile(qrdata);
    const result = await QrScanner.scanImage(qrdata);
    setData(result);
   };

   return <>
   <div className="checkBackground">
      <div className="checkContainer">
      <div className="heading">Counterfeit Product Detection</div>
        <div className="check">Scan a QRCode </div>
        <div className="body">
        <button type="button" onClick={handleClick} hidden="hidden">Scan QR code</button>
      <input
        type="file"
        id="file-upload"
        name="data"
        className="d-none"
        accept=".png, .jpg, .jpeg"
        ref={qrRef}
        onChange={retrieveFile}
      /><br />
```

```
      <div className="check-details">
        { data && (<p>Product Name: {data}</p>)}
        <button onClick={handleSubmit} >Get Details</button>
        { details && (<p>Details : {details}</p>)}
      </div>
        </div>
        <div className="footer">
          <button
            onClick={() => {
              setCheckOpen(false);
            }}
            id="cancelBtn"
          >
            Cancel
          </button>
          <button onClick={()=>{setCheckOpen(false);}} id="donebtn" >Done</button>
        </div>
      </div>
    </div>
  </>;
}

export default Check;
```

Pay.jsx

```
import { useState, useRef, useEffect } from "react";
import QrScanner from "qr-scanner";
import useEth from "../contexts/EthContext/useEth";
import "./Ship.css";
import "./Pay.css";

const Pay =({setPayOpen}) => {
   const [data, setData] = useState('');
   const [file,setFile] = useState(null);
   const [details,setDetails] = useState('');
   const qrRef = useRef(null);
   const { state: { contract, accounts }, web } = useEth();

   const handleClick =() => {
      qrRef.current.click();
   }

   const retrieveFile = async (e) => {
      const qrdata = e.target.files[0]; //files array of files object
      setFile(qrdata);
      const result = await QrScanner.scanImage(qrdata);
      setData(result);
   };

   const handleSubmit = async () => {
      const prodname=data;
      console.log(prodname);
      try{
        const name=sessionStorage.getItem("name");
        const eligible = await contract.methods.getLatestOwner(data).call({ from: accounts[0]});
     console.log(eligible);
     console.log(name);
     if(eligible==name && sessionStorage.getItem("addr")==accounts[0]){
        const receipt = await contract.methods.getPayDetais(prodname).call({ from: accounts[0] });
      console.log(receipt[0]);
      console.log(receipt[1]);
      console.log(receipt[2]);
      const name=sessionStorage.getItem("name");
      console.log(name);
      if(receipt[0]){
         setDetails("Payment to this product is completed");
      }else{
         setDetails("You can proceed with payment");
         console.log(details);
      await contract.methods.pay(prodname).send({
         from: accounts[0],
         value: web.utils.toWei(receipt[2],'ether')
       });
       //await contract.methods.pay(prodname).send({ from: accounts[0] });
       alert("Payment Successful");
       setPayOpen(false);
```

```
      }}else{
        setDetails("Current owner of product "+ data +" is "+eligible);
      }
      }catch(error){
        setDetails("No such product. It is Fake!!!!!!!!!!");
      }
    }

   return <div className="payBackground">
    <div className="payContainer">
    <div className="heading">Counterfeit Product Detection</div>
    <div className="pay">Pay for a product</div>
      <div className="body">
      <button type="button" onClick={handleClick} hidden="hidden">Scan QR code</button>
    <input
      type="file"
      id="file-upload"
      name="data"
      className="d-none"
      accept=".png, .jpg, .jpeg"
      ref={qrRef}
      onChange={retrieveFile}
    />
    <br/>
    <div className="check-details">
      { data && (<p>Product Name: {data}</p>)}
      <button onClick={handleSubmit} >Proceed to Pay</button>
       { details && (<p>{details}</p>)}
   </div>
   </div>
   <div className="footer">
          <button
            onClick={() => {
              setPayOpen(false);
            }}
            id="cancelBtn"
          >
            Cancel
          </button>
          <button onClick={()=>{setPayOpen(false);}} id="donebtn" >Done</button>
   </div>
   </div></div>;
}
export default Pay;
```

CHAPTER 6

TESTING AND RESULTS

6.1 TESTING

Testing is vital to the success of the system. System testing makes logical assumption that if all the parts of system are correct, the goal will be successfully achieved. This system is tested by following test cases and prepared for final implementation.

TEST CASES FOR COUNTERFEIT PRODUCT DETECTION USING BLOCKCHAIN

TEST CASE ID	TEST SCENARIO / DESCRIPTION	TEST STEPS	ACTUAL RESULTS	EXPECTED RESULTS	PASS/FAIL
[1]	Connection to MetaMask with ganache network	Connected to ganache accounts through MetaMask	Website opens	Website opens successful	Pass
[2]	Connection to MetaMask with other network Disconnection to MetaMask	Connection not established with MetaMask	Website not opens	Website not opens	Pass
[3]	Check user register with valid data and address	Go to website Enter all the details (email, name, password, user type, address) Click register	User should register into website	Registration successful	Pass
[4]	Check user register with already registered data	Go to website Enter all the details (name, email, etc.) Click register.	User should not register into website	Registration failed	Pass

	/ address				
[5]	Check customer login with valid data and account address	Go to website. Enter name, password. Click login.	User should login into website	Login successful	Pass
[6]	Check customer login with invalid data	Go to website. Enter name, password. Click login.	User should not login into website	Login failed	Pass
[7]	Check whether the website opens correctly to respective user type.	Once the user logins, it opens the respective user home page as specified at the time of registration	Website opens	Website should open	Pass
[8]	Check whether a product is added to a blockchain	Open manufacture module and click to button to add product. Enter the product details like name and cost. Submit the details.	Product added successfully	Product added successfully	Pass
[9]	Check whether the same product that exists in blockchain can add.	Open manufacturer module. Enter the same product that already exists and cost. Submit the details.	Product already exists	Product already exists	Pass

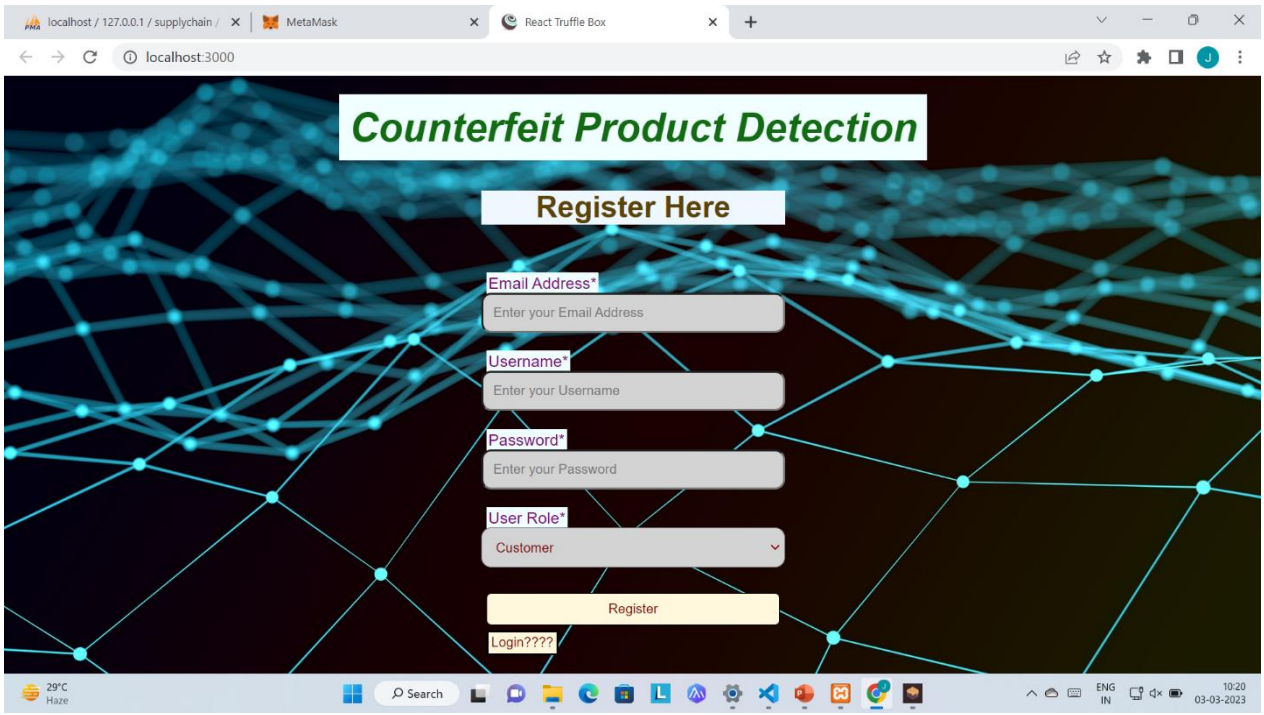
[10]	Check whether the payment gateway works	Open module other than manufacturer. Go to payment option and scan the product QR code and click payment option	Successfully purchased	Payment successful	Pass
[11]	Check whether the payment works only for the person that the product has shipped	Open module other than manufacturer. Go to payment option and scan the product QR code and click payment option	Displays the current owner of the product.	Displays the current owner of the product.	Pass
[12]	Check whether the shipment works for a person only after his/her successful payment.	Login to the website not as a manufacturer. Open the shipment module and try to ship the product that does not completed payment.	Complete your payment for the product to proceed with shipment	Complete your payment for the product to proceed with shipment	Pass
[13]	Check whether the shipment module works	Login to the website not as a manufacturer. Select the QR file of the product that needs to ship. Select the person to whom the product	Shipment successful	Shipment successful	Pass

		needs to ship. Click submit.			
[14]	Shipment access to other users that are not the current owner	Login to the website not as a manufacturer. Select the QR file of the product that needs to ship.	Displays the current owner of the product.	Displays the current owner of the product.	Pass
[15]	Check the details of a product and the ship track of it.	Login to the website. Select the QR code of the product to get all details about the product.	Displays all details about the product from manufacturer to retailer if exists with ownership and timestamp.	Displays all the details about the product from manufacturer to retailer if it exists.	Pass
[16]	Scan QR that does not exists in blockchain.	Login to the website. Select the QR code of the product to get all details about the product.	Displays “No such product. It is fake!!!!”	Displays “No such product. It is fake!!!!”	Pass

Table 6.1: Test Cases

6.2 RESULTS

Register Page



6.2.1 Figure 6.2.1: Register Page

Login Page

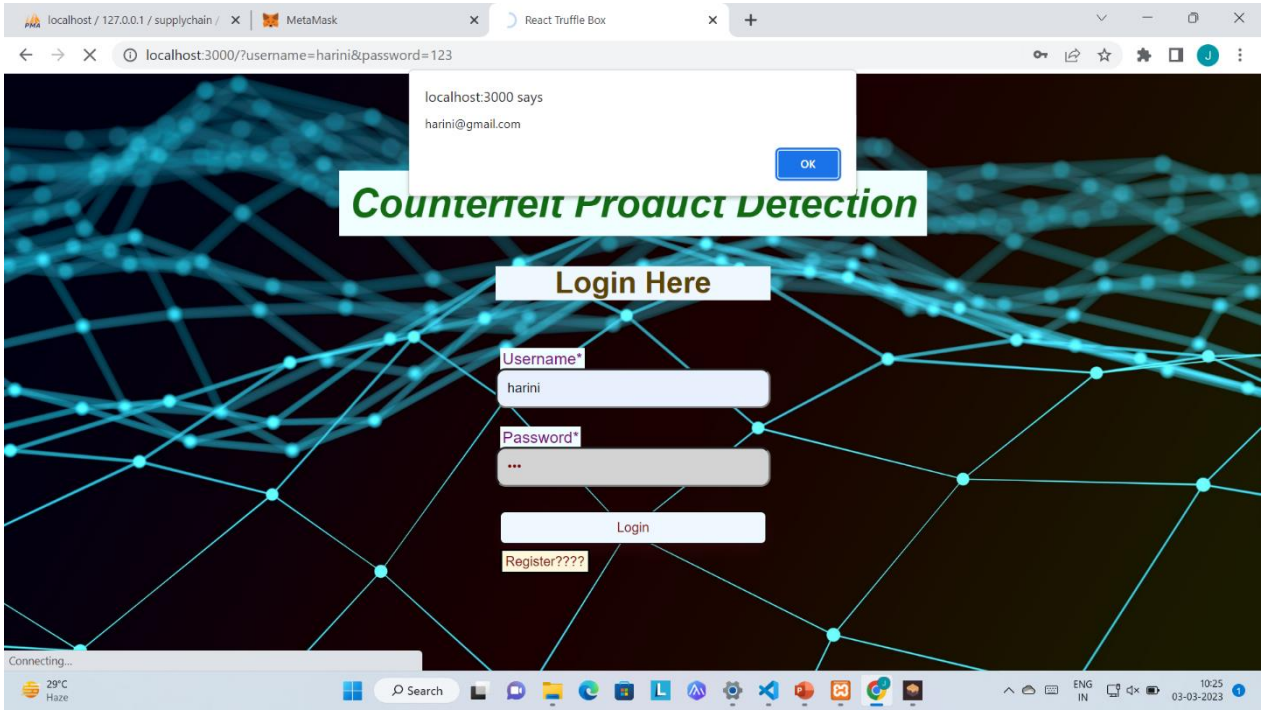


Figure 6.2.2: Login Page

Home Page of Manufacturer

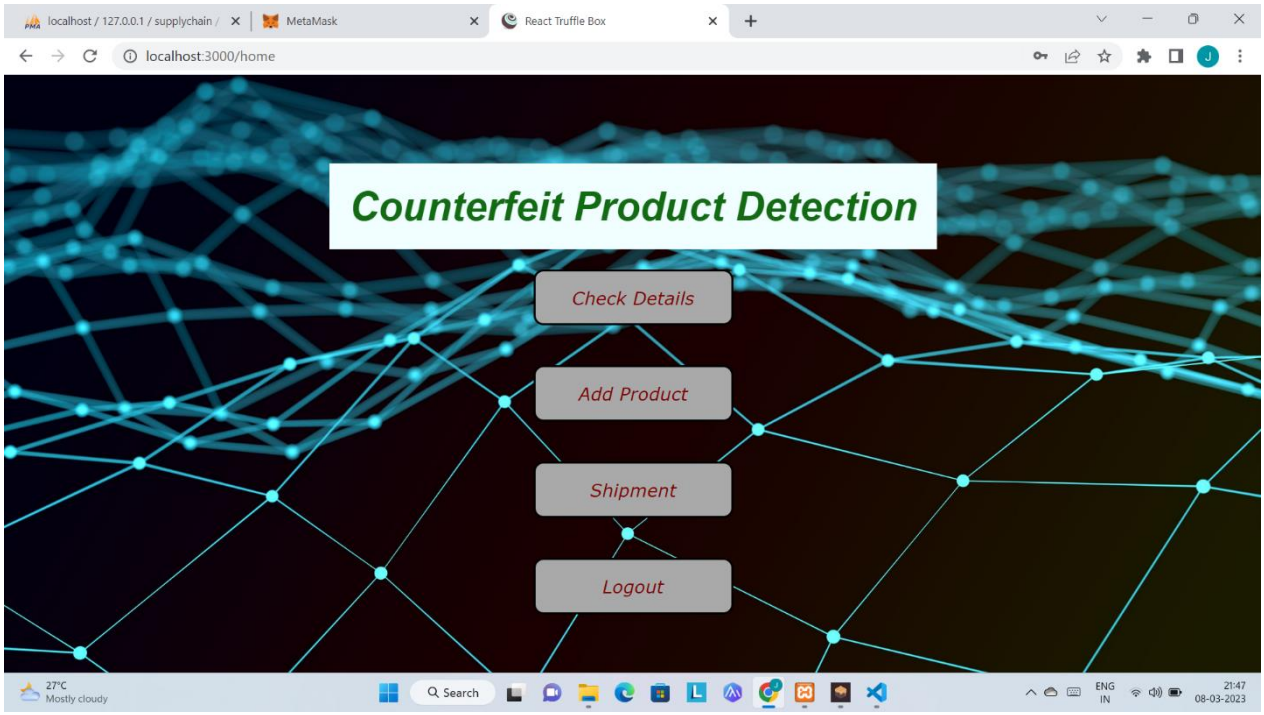


Figure 6.2.3: Home Page of Manufacturer

Add a Product

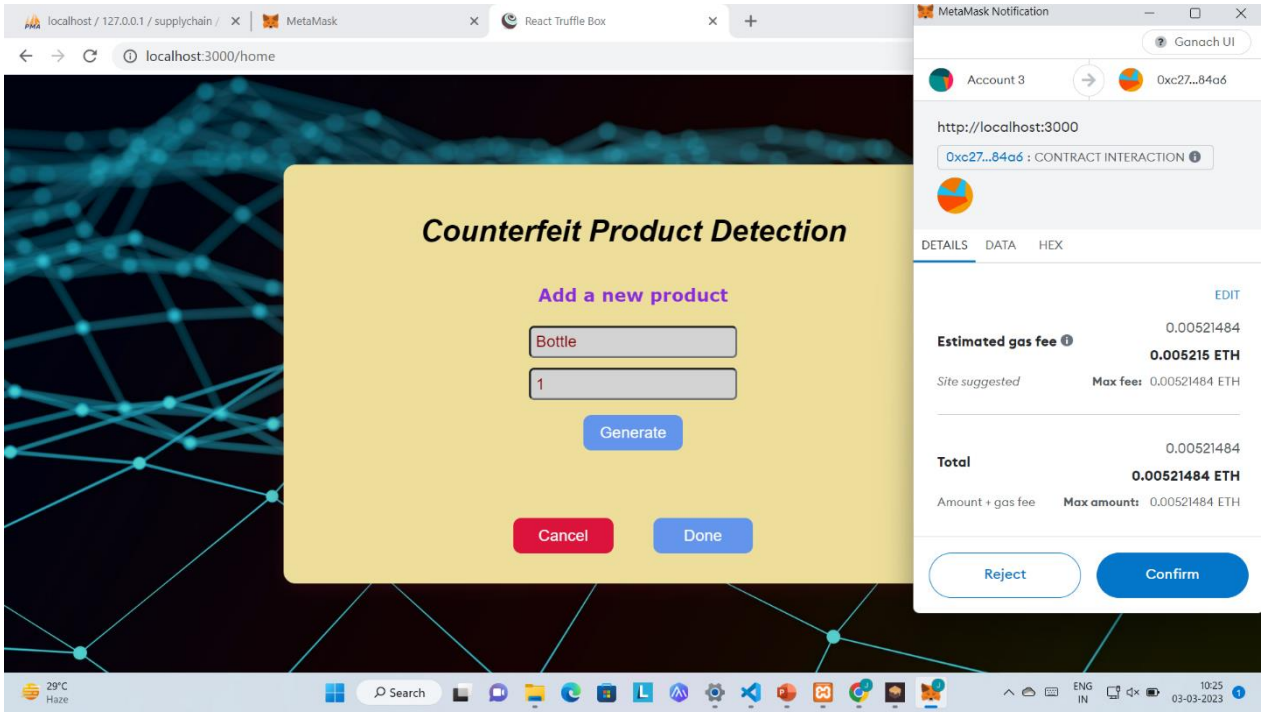


Figure 6.2.4: Page to add a product

Successful adding a product

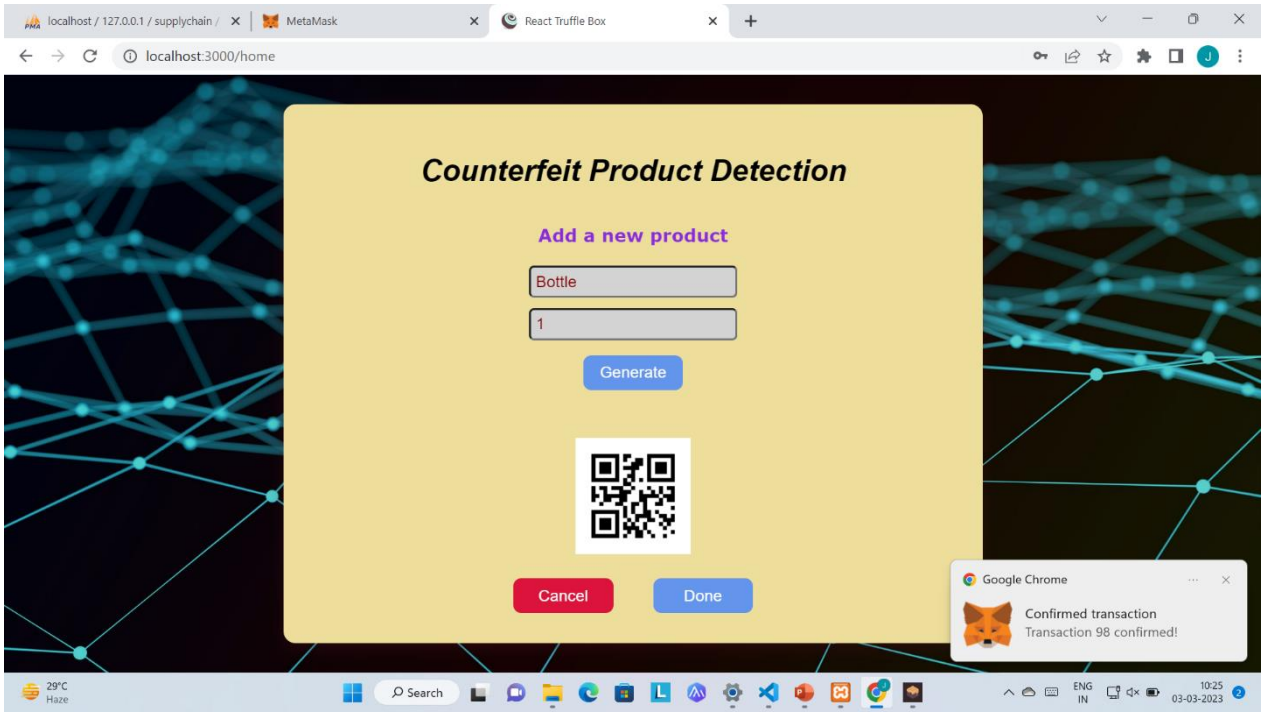


Figure 6.2.5: Successfully adding a product

Ship a Product

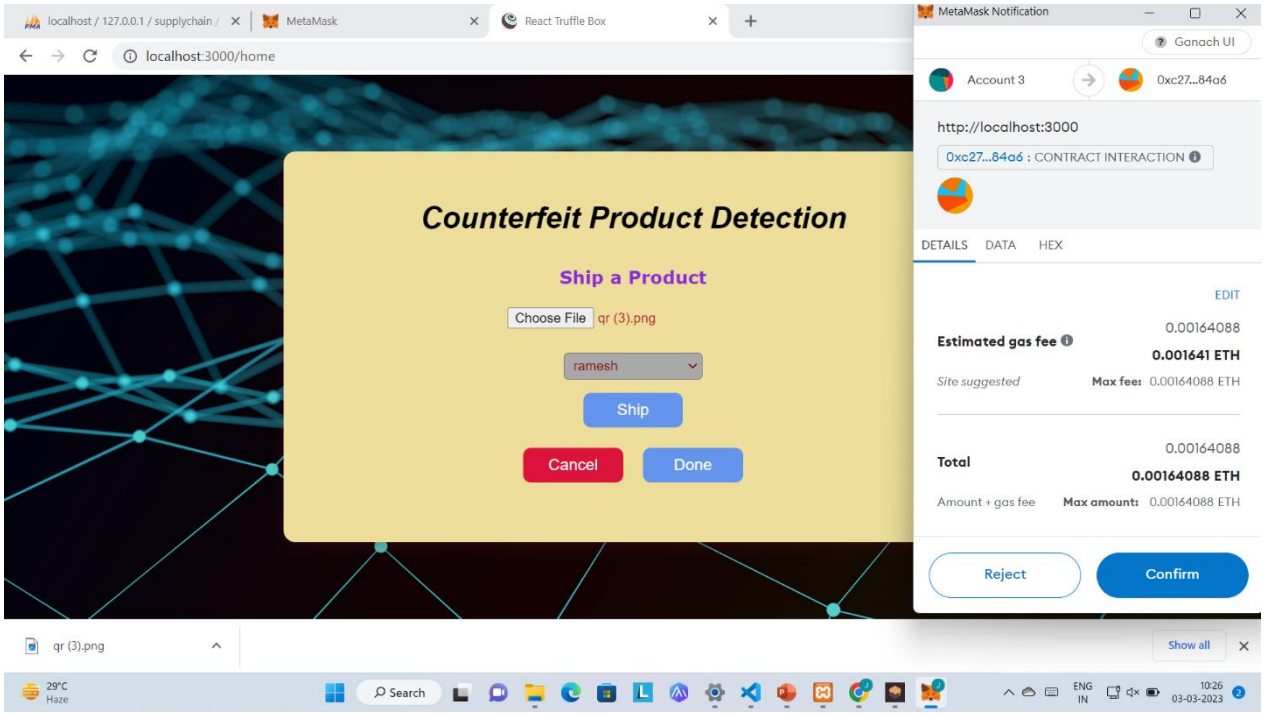


Figure 6.2.6: Page to Ship a product

Shipping Successful

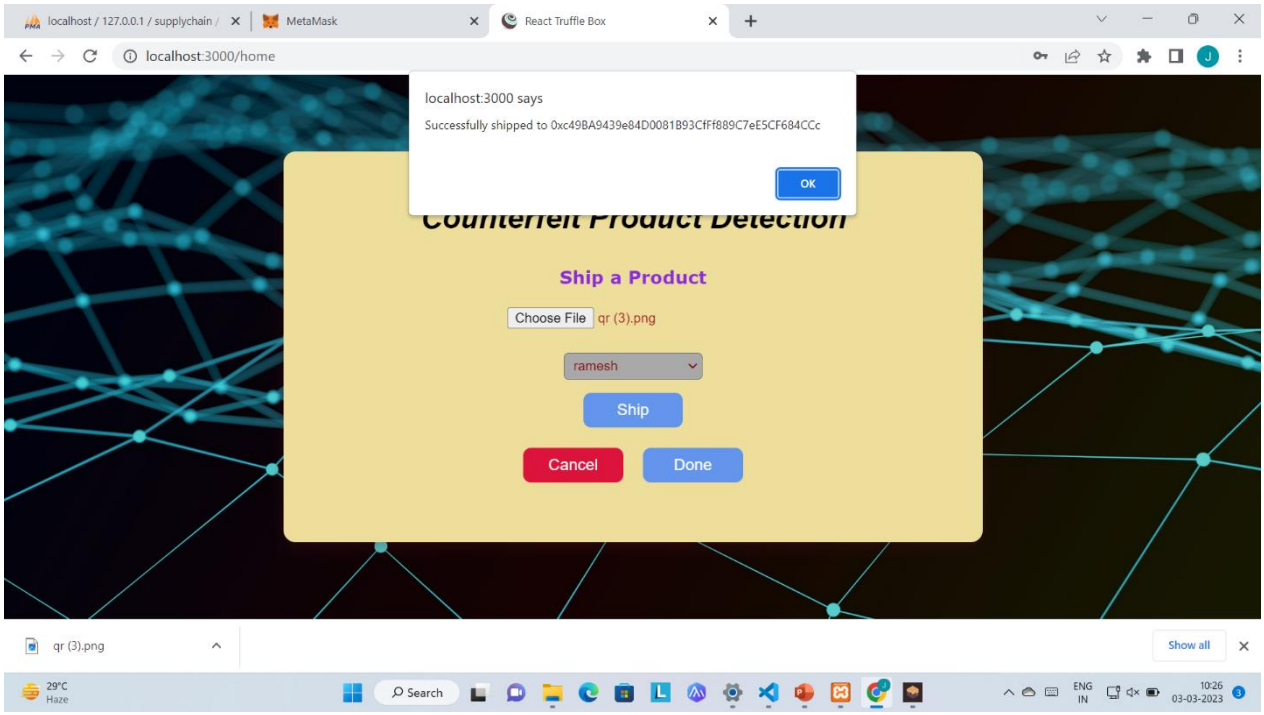


Figure 6.2.7: Shipment Successful Page

Home Page of Distributor / Retailer

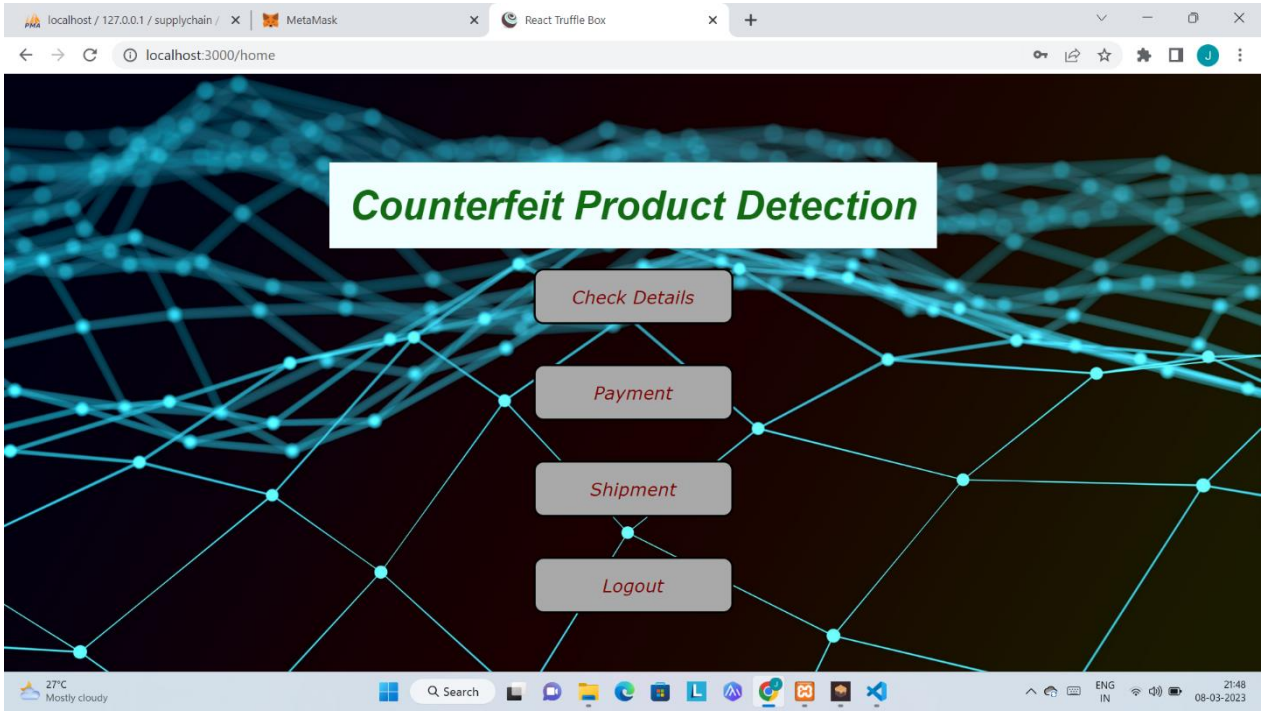


Figure 6.2.8: Home Page of Distributor/Retailer

Check Product Details

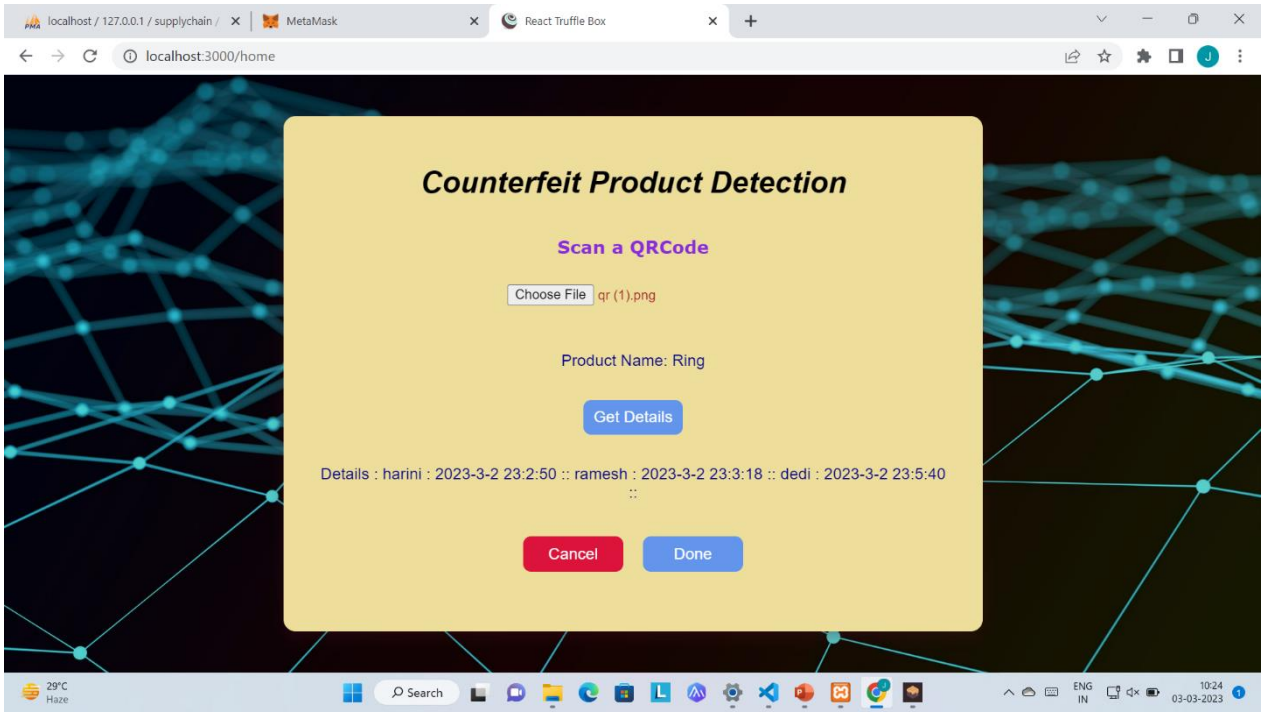


Figure 6.2.9: Check Product Details Page

Check Fake Product Details

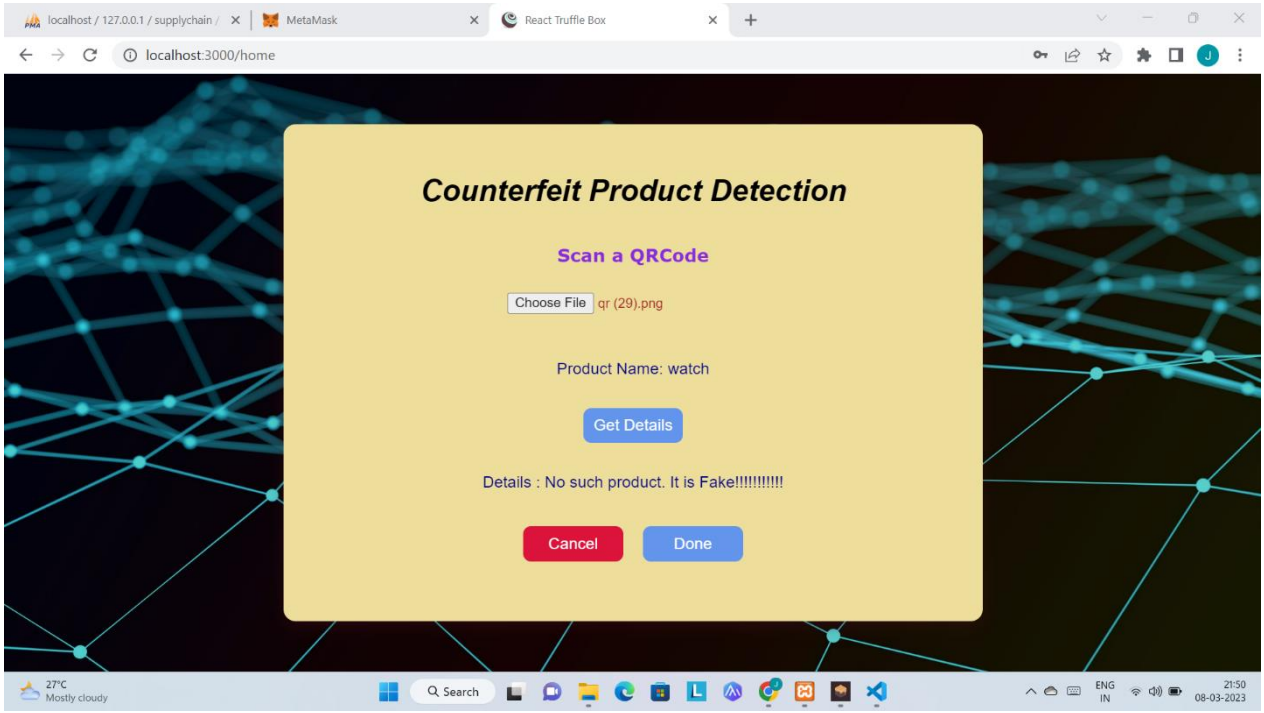


Figure 6.2.10: Detection of Fake Product

Shipment before Payment

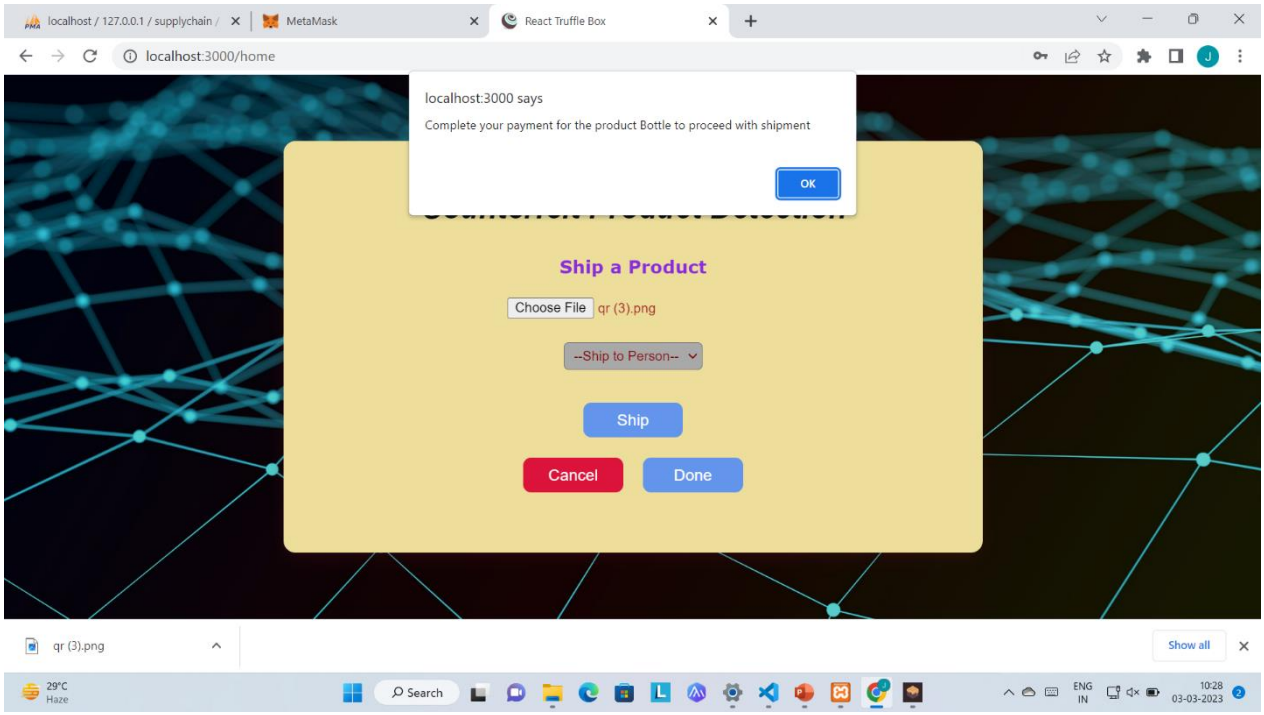


Figure 6.2.11: Shipment unsuccessful Page

Pay for a Product

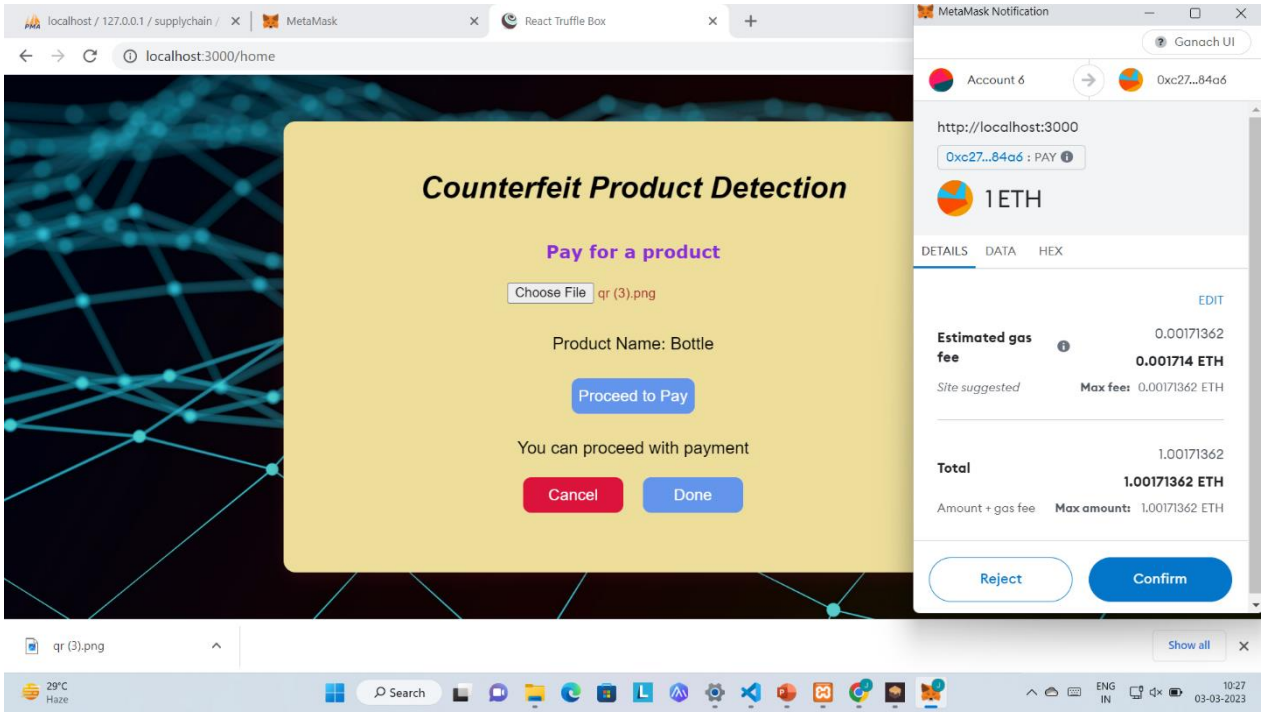


Figure 6.2.12: Payment Page

Payment Successful

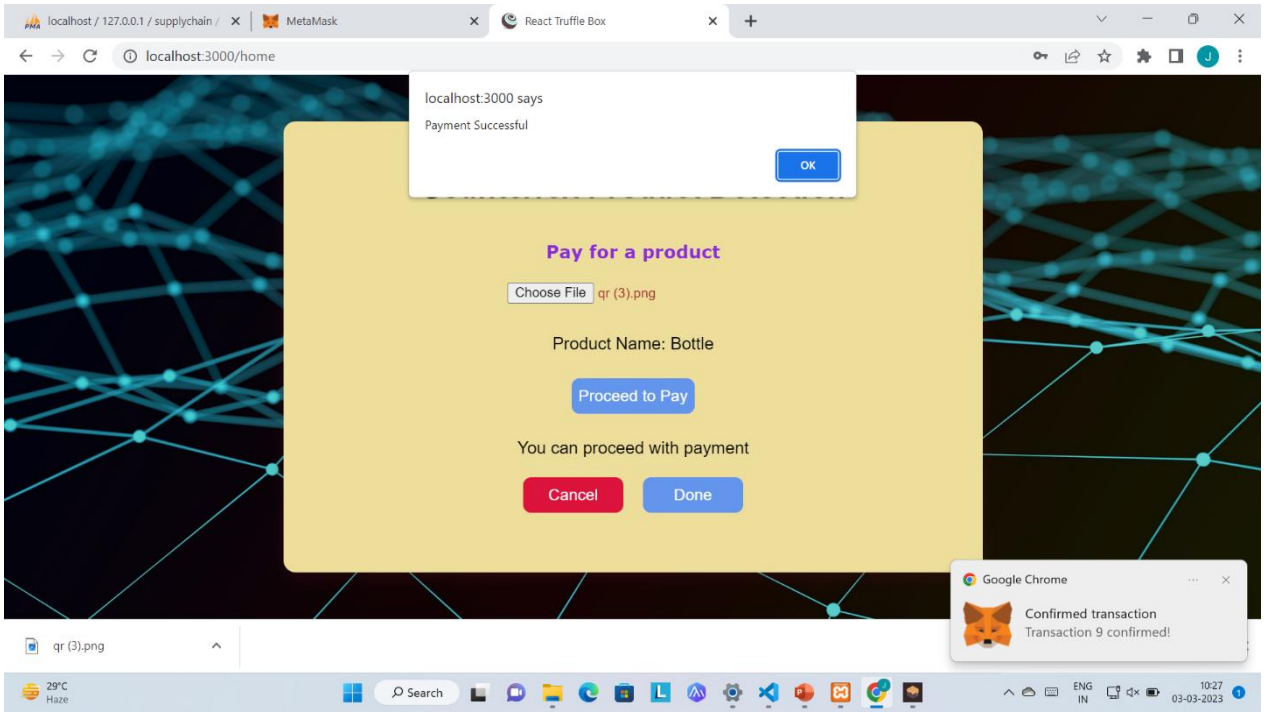


Figure 6.2.13: Payment Successful Page

Home Page of Customer

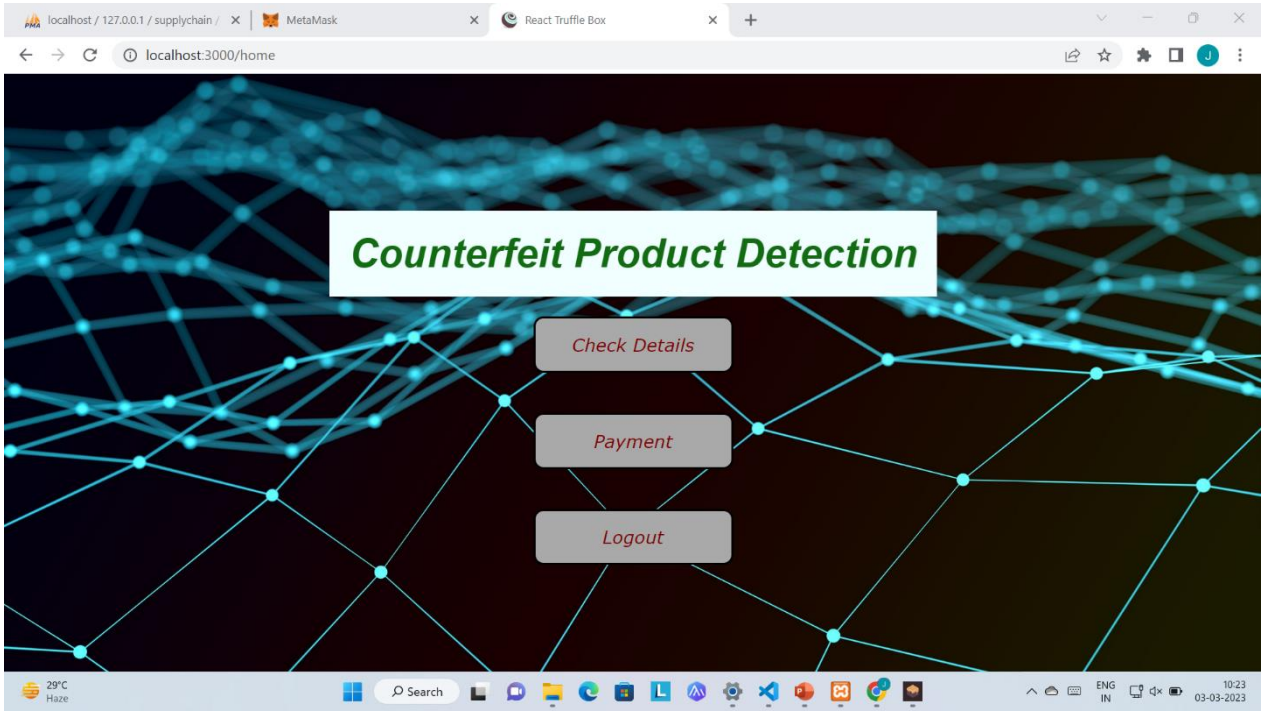
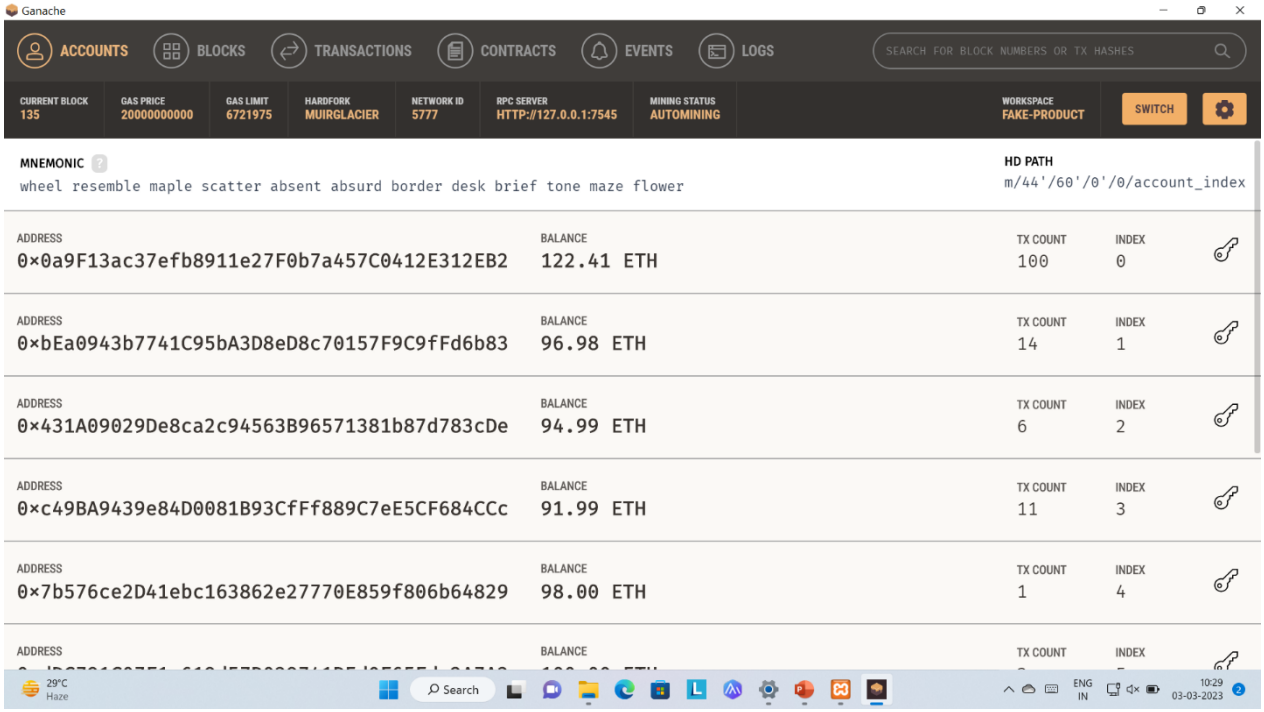


Figure 6.2.14: Home Page of Customer

Ganache Account Balances



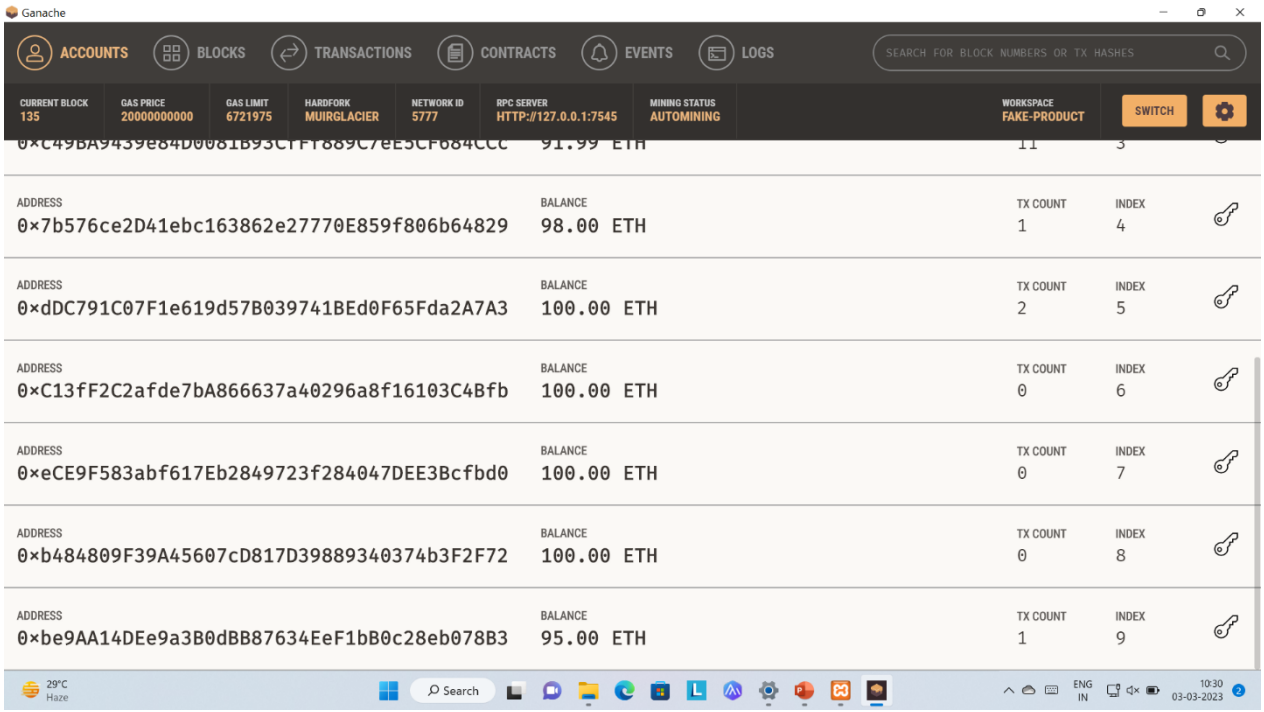


Figure 6.2.15,16: List of Ganache Accounts with Balances

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The blockchain technology for fake product detection has gained significant attention in recent years. Blockchain provides a tamper-proof and secure way to track products from their origin to the end user, making it difficult for counterfeiters to introduce fake products into the supply chain. By using blockchain, product information can be stored and shared in a decentralized way, ensuring the transparency and authenticity of the supply chain.

7.2 FUTURE SCOPE

- GST feature can be added to the application.
- Can de extended as a full-fledged E-commerce application.
- Product image, description and reviews on the product can be added.
- Product ratings can be added.

REFERENCES

Papers and articles

[1] Akhtar, et al. "A Blockchain-Based Solution for Product Counterfeiting in Supply Chain Management" (2021).

[2] Bayraktaroglu and M. O. Yıldız, et. al. "Fake Product Detection Using Blockchain Technology: A Review of Current Developments" (2021).

[3] Aini, Qurotul, et al. "Embedding a blockchain technology pattern into the QR code for an authentication certificate." Jurnal Online Informatika 5.2 (2020): 239-244.

[4] Ali, Omar, et al. "A comparative study: Blockchain technology utilization benefits, challenges, and functionalities." IEEE Access 9 (2021): 12730-12749.

[5] Al-Farsi, Sana, Muhammad Mazhar Rathore, and Spiros Bakiras, et al. "Security of blockchain-based supply chain management systems: challenges and opportunities." Applied Sciences 11.12 (2021): 5585.

[6] Bhutta, Muhammad Nasir Mumtaz, et al. "A survey on blockchain technology: evolution, architecture, and security." IEEE Access 9 (2021): 61048-61073.

[7] Daniel M. Hall and Hunhevicz, Jens J., et. al. "Do you need a blockchain in construction? Use case categories and decision framework for DLT design options." Advanced Engineering Informatics 45 (2020): 101094.

[8] Dursun, Taner, et al. "Blockchain Technology for Supply Chain Management." Global Joint Conference on Industrial Engineering and Its Application Areas. Springer, Cham, 2020.

[9] F. M. Castro-Mejía, et al. "Blockchain Technology for Supply Chain Traceability: A Literature Review" (2020).

[10] G. K. Mandal, et al. "Blockchain-based Framework for Anti-Counterfeiting in Food Supply Chain Management" (2020).

[11] J. J. Rodriguez-Molina, et al. "Blockchain-Based Anti-Counterfeiting Techniques for Next-Generation Pharmaceutical Supply Chain Integrity" (2020).

[12] J. Zhang, et al. "Fake Product Detection System Based on Blockchain Technology and Deep Learning" (2019).

[13] Jambhulkar, Swaroop, et al. "Blockchain-based fake product identification system." International Research Journal of Modernization in Engineering Technology and Science (2021): 2582-5208.

[14] Rajendran et al. "A Survey of Blockchain in Fake Product Detection" (2020).

[15] Shreekumar, T., et al. "Fake Product Detection Using Blockchain Technology." JOURNAL OF ALGEBRAIC STATISTICS 13.3 (2022): 2815-2821.

[16] Turjo, Manoshi Das, et al. "Smart supply chain management using the blockchain and smart contract." Scientific programming (2021).

[17] Wang et al. "A Blockchain-based Approach for Fake Product Detection in E-commerce" (2020).

[18] Wang et al. "A Novel Blockchain-based Traceability System for Anti-Counterfeiting in the Food Industry" (2019).

Web References

- Object Oriented Analysis and Design with UML, Charay Lerdsudwichai, Ph. D, Available: https://www.cpe.ku.ac.th/~jan/214574/review.pdf
- Static Modeling using the Unified Modeling Language, Spiros Mancoridis, [Online] Available: https://www.cs.drexel.edu/~spiros/teaching/CS575/slides/uml.pdf
- Metamask Inc, Metamask, Website https://Metamask.Io/,.
- DOCUMENTATION. (n.d.). Retrieved from Truffle Suite: https://www.trufflesuite.com/docs.