Tribhuvan University
Institute of Science and Technology
A Final Year Project Report
On

"Product Identification using Blockchain Technology"

Under the supervision of

Er. Prabin Kumar Jha

Submitted to:

Tribhuwan University Institute of Science and Technology

Submitted by

Rajesh Pudasaini (15980)

Prabesh Oli (15977)

Ujwal Dhital (16000)

Department of Computer Science and Information Technology Madan Bhandari Memorial College

Binayaknagar, New Baneshwor, Kathmandu, Nepal

April 2022

RECOMMENDATION

This is to recommend that **Rajesh Pudasaini (15980/074), Prabesh Oli (15977/074), and Ujwal Dhital (16000/074)** have carried out project work titled **"Product Identification using Blockchain Technology"** for the fulfillment of the **Bachelor of Science in Computer Science and Information Technology** under my supervision. To my knowledge, this work has not been submitted for any other degree and is ready for theFinal Defense presentation.

They have fulfilled all the requirements laid down by the Madan Bhandari Memorial College, Department of Computer Science and Information Technology, Anamnagar, Kathmandu.

……

Er. Prabin Kumar Jha

Project Supervisor

Department of Computer Science and Information Technology,

Madan Bhandari Memorial College

Anamnagar, Kathmandu, Nepal

Madan Bhandari Memorial College

Affiliated with Tribhuwan University

CERTIFICATE OF APPROVAL

The undersigned certify that they have read and recommended to the Department of Computer Science and Information Technology, IOST, Madan Bhandari Memorial Campus, a project report entitled **"Product Identification using Blockchain Technology"** submitted by Rajesh Pudasaini(15980/074), Prabesh Oli (15977/074), and Ujwal Dhital (16000/074). The project was carried out under special supervision by Er. Prabin Kumar Jha and within the time frame prescribed by the syllabus.

We found the students to be hardworking, skilled, and ready to undertake any related work to their field of study and hence we recommend the award of partial fulfillment of a Bachelor's degree in Computer Science and Information Technology.

Signature of Supervisor **Name:- Er. Prabin Kumar Jha** **Designation:-** **Signature:-** **Date:-**	**Signature of HOD/Coordinator** **Name:-** **Designation:-** **Signature:-** **Date:-**
Signature of the External Examiner	**Signature of the Internal Examiner**

Acknowledgment

We would like to offer our most sincere appreciation to Madan Bhandari Memorial College and Tribhuwan University for allowing us to carry out this project work for academic and professional improvement. We are very grateful to our teachers and professors who assisted us throughout the different phases of this project. The valuable guidance and insights provided by **Mr. Prabin Kumar Jha** - our project supervisor especially played adefining role in the successful completion of this project.

We would also like to acknowledge respected coordinator **Mr. Phul Babu Jha** along with all the teachers of the Madan Bhandari Memorial College who were directly or indirectly monumental in the succession of this project.

Finally, we are also thankful to our parents for always being there for us in every possible situation during this important work of our final year in college.

Abstract

Product Identification is an essential model that allows the seller to add a product to a decentralized platform such as Blockchain and allows buyers to purchase the product from the decentralized platform. Fraud products, counterfeiting, and duplication are the current marketplace's major problems. This project is a step forward in verifying the product identification with their information, ownership, and validity detail. This system maintains the buyers, sellers, and product details in a decentralized blockchain platform such as Ethereum. This report details the entire project development process from planning, analysis, design, implementation, and testing. Verifying the product ownership and its information to get the original product is the major difficulty in this space, but this project systematically solves some of those problems. This project is an improvement of the current centralized way of purchasing goods online, where the information remains as it is entered by the seller while listing the product.

Keywords: Decentralize platform, Security, Product uniqueness, Product valuation

Table of Contents

List of Figures

List of Tables

List of Abbreviations

JS	JavaScript
API	Application Programming Interface
CSS	Cascading Style Sheet
GUI	Graphical User Interface
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
UI	User Interface
UML	Unified modeling language
NFT	Non-Fungible Token
RFID	Radio Frequency Identification
DApp	Decentralize Application
SQL	Structured Query Language
ECDSA	Elliptic Curve Digital Signature Algorithm

Chapter 1: Introduction

1.1 Introduction

Production Identification is the platform through which people can incorporate into a business strategy around the globe. Product Identification creates a marketplace for the people who want to sell and buy the product in a secured manner, as this system is based on blockchain technology such that it ensures the proper security of the data of the user who are using this system. Counterfeiting and duplication are always a problem when a product or technology is exported, as they can harm a company's brand, revenue, and customer health. Counterfeit or fraudulent items have become a significant cause of concern for producers, resulting in insignificant losses. This is where the Product Identification system comes into action as it is using blockchain technology, which can be used to verify the product's credibility.

A blockchain is a distributed network of personal computers that keeps track of duplicated transactions in a digital ledger. Every block contains many trades, and whenever a new transaction is conducted on the Blockchain, it is logged in all participants' records. Blockchain is a distributed ledger technology that records transactions using a hash, an unchangeable cryptographic signature. Counterfeiting is a problem that blockchain technology may be able to help with.

With this system, the Buyer will have assurance as the product purchased from a Customer or a Brand is genuine and its value to its age. Within this system, the seller will have the option to enter the detail about the product whether it's an old or new product. The basic information about the product such as name, and images will be stored in the normal database but the actual information such as manufacturing date, Batch No, and Barcode mapping is stored in Blockchain which is not editable as a chain of blocks. Also, no intruder can interfere with it as the smart contract will be deployed in a secure Ethereum platform. Now once sellers publish the ads for their product, buyers will search for it and make their choice. The buyer now will have the option to view the product detail if the seller allows anyone to view it, else the buyer will contact the seller and asks for the product detail. After the buyer will make the purchase, the seller will transfer the ownership to the buyer, and a new transaction log is saved in a blockchain.[1]

1.2 Problem Definition

Fake Product Identification is the major problem our system is focused on. Counterfeiting and duplication are always a possibility when a product or concept is sold globally. Fake documents could endanger the company's reputation as well as the security of its customers. Detecting a counterfeit item is currently the most difficult challenge. As a result, item creators will face significant challenges. In the countries like Nepal and India,fake and counterfeit products are an issue. Using Blockchain technology, the proposed method creates a unique transaction ID. Trade records are preserved in blocks in this method. It is difficult to change or obtain access to the information stored in these blocks since it is protected.

Another problem identified is price value fluctuation. The more seller uses that product, the less its value will be going to be. But not in the scenario of Nepal, as we don't have any source to verify when this product was purchased first. And how many times the product is resold.

These are the major problems identified for both the customers and the renters in the vehicle renting system in our country and the proposed system will set its objectives to alleviate both these problems.[2]

1.3 Objectives

i. To identify the detailed information of the product such as (product owners, manufacturing date, and purchasing date), before the buyer makes any transaction with the seller

ii. To value the product price based on its age.

iii. To track the transaction that is made between buyer and seller.

1.4 Scope and Limitations

Our long-term goal for this platform is to make it simple for people to purchase and sell items, as well as to pay the right price for them and to sell them at a price that is appropriate for their value. Furthermore, this project aims to educate people about Blockchain, one of the most prominent technologies, and its application in the field of themarketplace. The global e-commerce market is expected to total $5.55 trillion in 2022.

Some of the limitations that can hinder this application are:

1. The cost to gas fee is too costly

2. Adopting blockchain technology in the current marketplace is not easy as the term itself is not familiar enough.

3. Government is also not ready to bring blockchain technology, so its implication into the market is not as easy as implementing other platforms.

1.5 Development Methodology

The agile software development methodology is a process for developing software. Agile means the ability to move quickly and easily and respond swiftly to change this is a key aspect of Agile software development as well. Extreme Programming (XP) is an agile software development framework that aims to produce higher quality software, where software is developed in incremental cycles. It is used for time-critical applications, and for dynamically changing software requirements. It is suitable for a small co-located extended development team.

In the development of the project, blockchain technology was studied in detail. In the initial phase of the project, current blockchain technology, which is being used around different countries in the world, was studied. Considering the usefulness of those systems, the requirement of our system was defined. Our study on the current product identification process in Nepal showed that an online purchasing system through which people could buy or sell products without having to go to the marketplace is needed to increase security and fairness. A good buying and selling of products must have security, reliability, dynamic, and un-editable at the same time. To satisfy all these criteria, blockchain technology was chosen and different articles and papers were studied. After defining the requirements, a smart contract was written in Solidity.

It was deployed and tested using Remix ide. It was checked if the smart contract could meet all the requirements. In this initially deployed smart contract, every address could buy and sell the products. This smart contract was again tested and was modified till it could meet all the functional and non-functional requirements. All the change in the smartcontract was incorporated into the backend. To test Hardhat network was used.

The proposed system requires rapid change. For the dynamically changing software

requirements, XP is appropriate and it was chosen for this project. This model also reduced the risk caused by the fixed-time project using new technology and thus the final project could be delivered in time. Solidity and metamask being new technologies were unstable and to adopt the changes, the agile development model was the best. It allowed to break lengthy processes, automated units, and functional testing and thus helped to deliver the project in time.

1.6 Report Organization

Chapter 1: Introduction

This chapter explains the overview, introduction, problem statement,objectives, scope, and limitations of the proposed system.

Chapter 2: Background Study and Literature Review

This chapter covers the background study and the literature review of theproposed system.

Chapter 3: System Analysis

This chapter covers all the history, methods, requirement specifications andfeasibility, and structured system requirements.

Chapter 4: System Design

Design of Product Identification Using Blockchain is explained in detailwith all the necessary diagrams and brief functionally.

Chapter 5: Implementation and Testing

The process of implementation and testing is described along with all the toolsused for the development.

Chapter 6: Conclusion and Future Recommendation

The conclusion and future scope of the application are explained.

Chapter 2: Background Study and Literature Review

2.1 Background Study on Product Identification using Blockchain

Blockchain is an emerging and robust tool that consists of distributed (i.e., without a single repository) and usually decentralized digital ledgers that are tamper obvious and resistant to tampering (i.e., a bank, company, or government). At their most basic level, they allow a group of users to record transactions in a shared ledger within that group, with the result that no transaction can be modified once it has been published, as long as the blockchain network is operational. Our report/project is essentially a blockchain technology application. Product identification is a broad labeling category that encompasses product traceability, brand protection, and numerous information labels. Product identity labeling is vital in today's fast-changing business environment, where theft and counterfeit products are constant dangers. Blockchain is a relatively new concept in Nepal, as well as an emerging topic globally. In and of itself, product identification is insufficient. It is insecure and prone to data fraud. As a result, Product Identification with Blockchain decentralizes data, enhancing data security and transparency for users.

2.2 Literature Review

"The blockchain is an incorruptible digital ledger of economic transactions that can be programmed to record not just financial transactions but virtually everything of value." The use of blockchain reached another milestone when West Virginia became the US's first state to allow internet voting by blockchain in the primary elections. In an interview, Votem CEO Pete Martin said, "Blockchain technology provides all of the characteristics you would want in a platform that is arguably the most important part of the democratic society".

As the era of Digitization is growing, there is a maximum chance of being the victim of a fake Digitization world. One of them is a Fake Product. The study of previous research can be valuable as it provides insights into different systems that have been developed for fake product identification and also the limitations of such systems are studied. one of them is Fake Product Detection using Blockchain Technology (2021) by Tejaswini Tambe and team, where they relate the relation from direct Manufacture to the customer as a supply

chain management. And customer on their end confirms as it is a genuine product and manufactured by the specific manufacturing brand. Zhu et al proposed a system in their paper that prevents counterfeiting of drugs and increases the traceability in the pharmaceutical industry using Blockchain Technology. And in support of this system Sahoo et al proposed the ECDSA(Elliptic Curve Digital Signature Algorithm) in their paper which added traceability and visibility for the protection of drug counterfeiting in the pharmaceutical industry. Similarly, Sun et al. developed an Ethereum and Radio- frequency identification (RFID) based system modeling combined with the Blockchain and Internet of Things (IoT) technologies to prepare a fake liquor detection system. Therefore previous researches point out that there is direct chaining of the manufacturer tothe customer and is related to the specific area such as medicine which could be better ifit relates in general.

Also, in the context of Nepal, HamroBazaar is one of the popular e-commerce sites for C2Cbusinesses. What it lacks is the trust in a product of a specific person or brand who is selling it. With their platform, we can get the old as well as a brand new product within a minute of buyer and seller contact. They priorly focus on building customer-to-customer and business-to-customer relationships. One is selling the product at a specific price and the other is buying at the same price with slight bargaining.[3]

Chapter 3: System Analysis

3.1 System Analysis

Analysis of the system was carried out so that there is a strong guarantee of the system's utility and to facilitate the smooth completion of the project. The system analysis contains information about the requirements of the system as well as a feasibility study of the project.

3.1.1 Requirement Analysis

Requirement analysis is carried out so that the functional and non-functional requirements of the system can be figured out and implemented accordingly. The outline of the functional and non-functional requirements is shown below.

3.1.1.1 Functional Requirements

1. There shall be an option for the seller to add numbers of the product (old + new) in our system. There is no barrier to limiting the product count.
2. The buyer shall view the product without any authentication mechanism.
3. The system shall store the user product detail in a decentralized ledger.
4. The buyer shall view the purchased items.
5. The seller shall view the list of products, that is listed for sale.

Use Case Diagram

The given diagram shows a generally high-level flowchart of the Product Identification System. The diagram can be used to understand the general working principle of the system. The process starts with a registered user logging into the system. The homepage will provide the user with options to either act as a buyer or a seller. If the user is a seller, s/he can post a product ad to the platform. The users who are registered as a seller can also be a buyer. Once the buyer is ready to purchase a product from the seller, the seller will have an option to transfer the ownership and a log is maintained in a Blockchain.

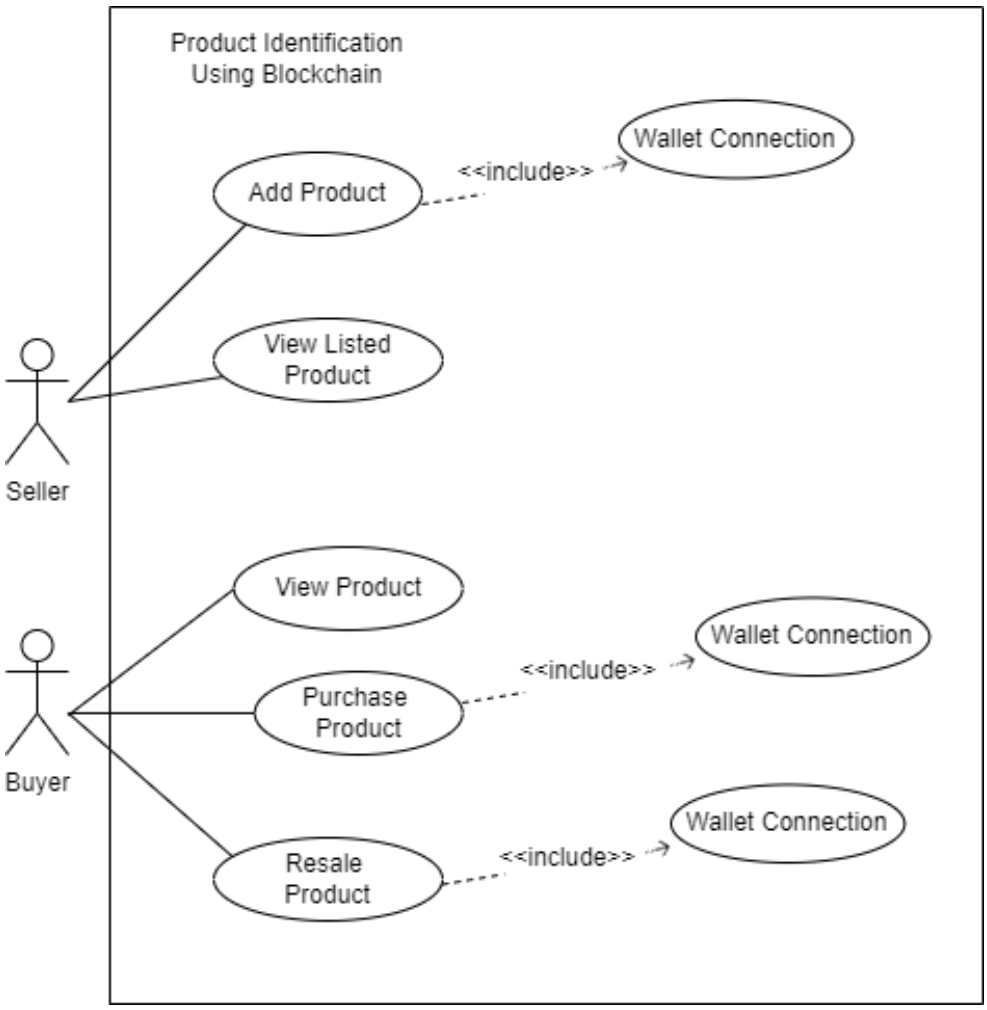


Figure 2: Use Case Diagram

3.1.1.2 Non-Functional Requirements

1. The UI should be clean and intuitive.
2. A proper wallet connection should be established.
3. The buyer shall view the product detail along with its previous owners and dates

3.1.2 Feasibility Analysis

3.1.2.1 Technical Feasibility

The proposed system requires technical resources which mostly include basic hardware and software. All of the members in the group have laptops and the necessary hardware that will be required in project development. In terms of software required for the development, the availability of Integrated Development Environment is also not of concern as they are readily available.

To develop, the front end, “React”, “JavaScript” is used, “decentralized ledger” is used for database-related aspects of the system whereas “Solidity” is considered for developingthe smart contracts of the system. All of these languages and software are available to the group members. Moreover, collaboration using social media can be done and since all of the members are connected to the internet, this portion will not be that complex either. Therefore, under normal circumstances, the proposed system is technically feasible as the hardware, software and necessary elements will not be compromised during the project’s development.

3.1.2.2 Operational Feasibility

Operational feasibility measures the extent to which the system will solve the problem that is identified and fulfills the requirement of the potential users. In the current scenario,there is a lot of demand in Nepal regarding the product marketplace. The healthy boom in e-commerce and online product trade increases the need for a security marketplace. The points were taken for the operational feasibility of the proposed system.

1. The proposed system will likely solve the secure marketplace problem for business operations by providing a secure platform that has been built using blockchain technology.
2. The proposed system keeps the track or records of any products maintaining the genuinity of the products and providing information about the products showing how many times they have been resale.
3. The proposed system also ensures the safety of the data of the users of this system.
4. The proposed system is using Metamask to connect user wallet and allows to trade the products in market place.

3.1.2.3 Economic Feasibility

Feasibility determines if the project goal can be achieved within resources limit allotted to it or not. Since the proposed system has been built using blockchain technology such that there need to pay the gas fee to carry out every transcation and products price are also traded in crypto such that following considerations are taken for economic feasibility.

1. The proposed system is using smart contract for memory management such that it can minimizes the economic burden to accomplish this project to some extent.
2. Since this is college project so, the proposed system is deployed locally using Hardhat (Local Ethereum Network) in public network due to which there is no any kind of deployment charge.

3.1.2.4 Schedule Feasibility

The task of completing the project is broken down into different phases such as analysis, and design. Coding, Testing, Implementation, and Documentation. The documentation step continued from beginning to end whereas the rest of the actions were completed as shown in the Gant Chart.

It took nearly 5 months for the completion of the project which can be seen in the Gantt Chart bellow and Trello tool screenshot is provided in the Appendix that informs about the scheduling of the project.

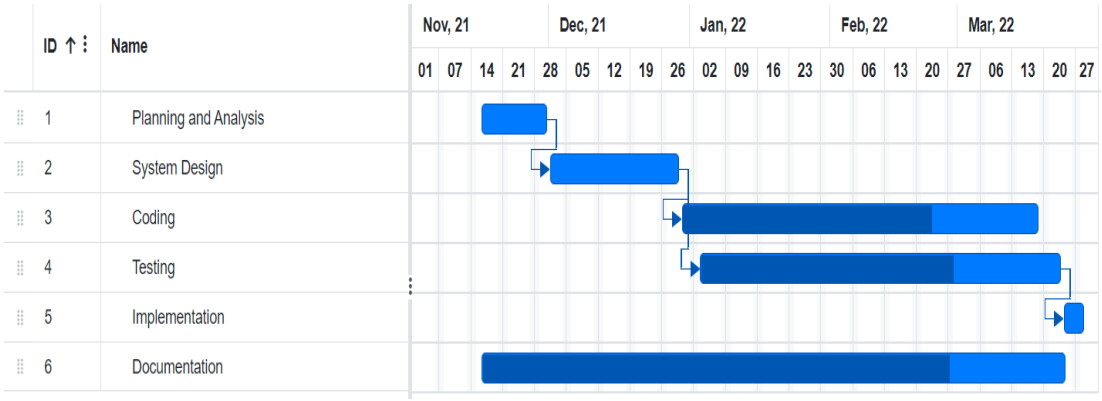


Fig: Gantt Chart

3.1.3 Analysis

3.1.3.1 Acitvity Diagram

The given diagram shows a generally high-level activity diagram of the Product Identification System. The diagram can be used to understand the general working principle of the system.

The process starts with a wallet connection to the system. The homepage will provide the user with options to either act as a buyer or a seller. If the user is a seller, s/he can post a product ad to the platform. The users who are registered as a seller can also be a buyer. Once a buyer is ready to purchase a product from the seller, the seller will have an option to transfer the ownership and a log is maintained in a Blockchain.

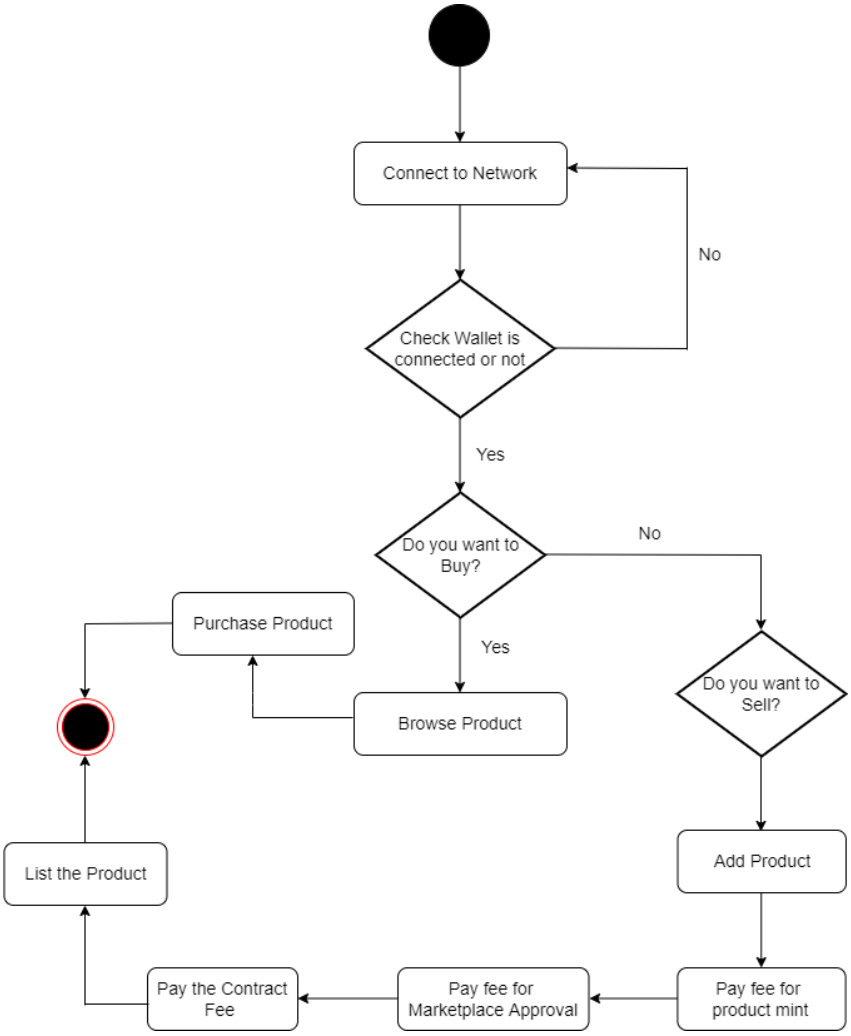


Figure 1: Flow Chart Diagram

3.1.3.2 Class Diagram

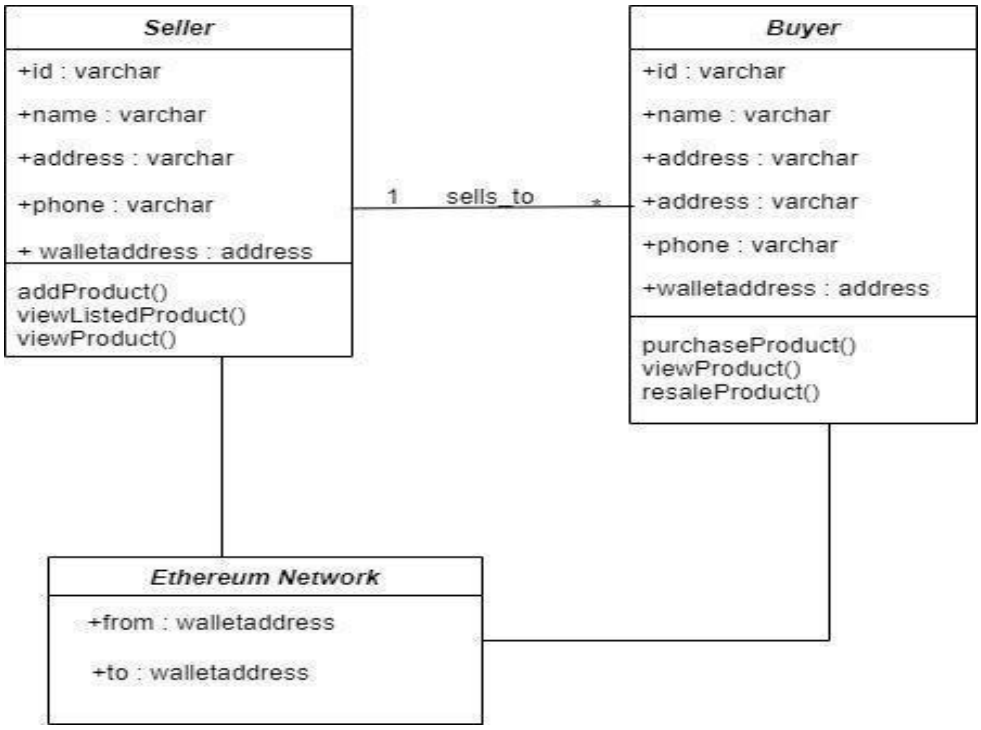


Figure 3: Class Diagram

The given class diagram shows the attributes and functions of the actors in the system. The relationship between seller and buyer is 1 too many meaning that one seller can have many buyers. Similarly, an admin can manage many systems and users.

3.1.3.3 Sequence Diagram

The sequence diagram is used primarily to show the interaction between objects in the sequential order that which that interaction occurs.

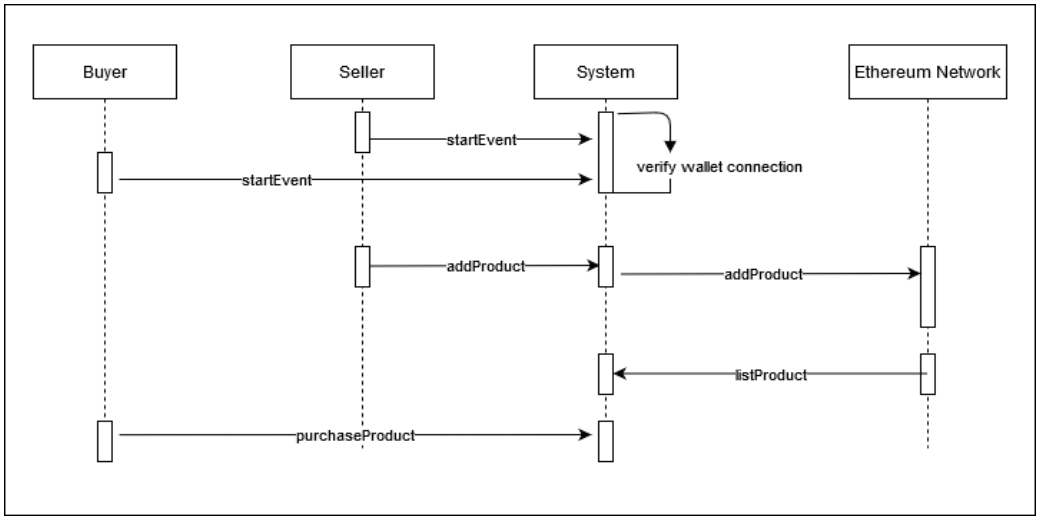


Fig: Sequence Diagram

Chapter 4: System Design

4.1 Design

System design is the process of defining the elements of a system such as the architecture, modules, and components, the different interfaces of those components, and the data that goes through that system. A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and views of a system. After the requirement analysis, the system is designed and the process of system design is represented by a highly detailed activity diagram and a database schema that is used for making the application.

4.1.1 Component Diagram

4.1.2 Deployment Diagram

The deployment diagram shows the execution architecture of the system. Here the proposed system execution starts when the Solidity compiler (solc) compiles the smart contract written in solidity. Once the Smart contract gets compiled by solc the Smart contract gets converted into Application Binary Interface (ABI) and bytecode as shown in figure below. ABI is in high level form such that our system interacts with Smart contract using that ABI and the bytecode is deployed into the Ethereum Blockchain.

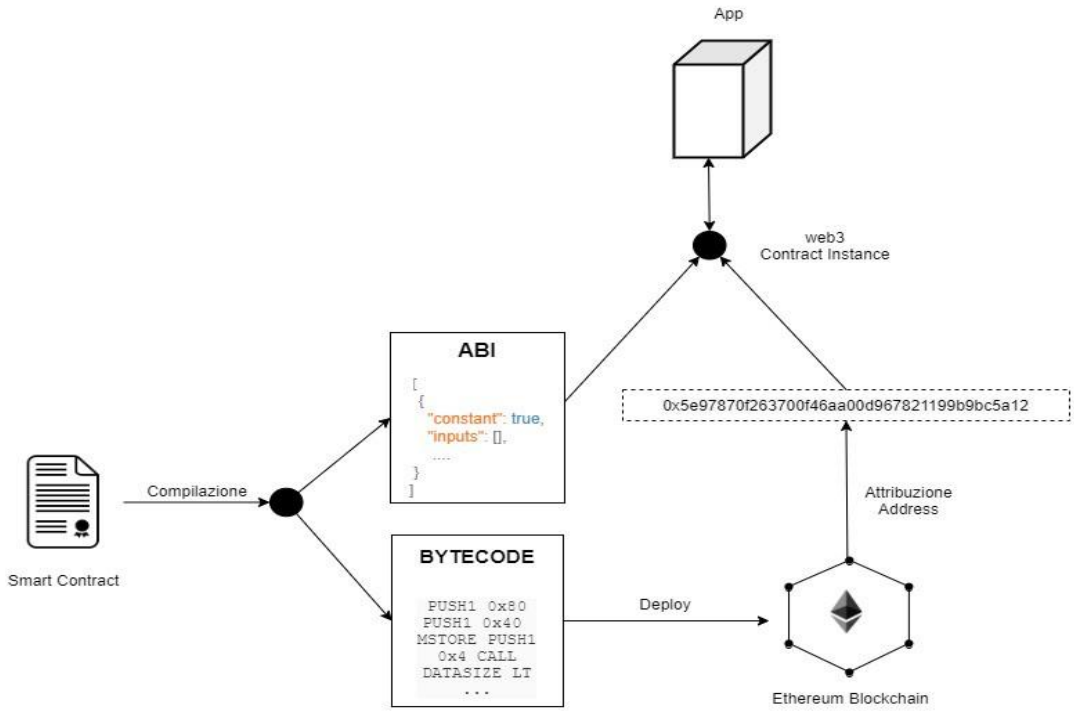


Fig: Deployment Diagram

4.2 Algorithm Details:

Here are the steps that are followed in order to deploy our smart contracts in the Ethereum Blockchain network and interact with it.

1. **Compile the Smart Contracts:**

Smart contract is written in Solidity programming language. So, smart contracts were compiled with Solidity compiler (solc).

solc Marketplace.sol

Returns bytecode as

0x60c060405234801561001057600080fd5b5060405161120b38038061120b83398181016
0405281019061003291906100945 65b6001600081905550 3373ffffffffffffffffffffffffffffffffff
ffffff1660808173ffffffffffffffffffffffffffffffffffffffff1660601b815250508060a081815250505
06100de565b6000815190506100 8e816100c7565b92915050565b6000602082840312156100
0a657600080fd5b60006100b48482850161007f565b9150509291505056 5b6000819050919
050565b6100d0816100bd565b81146100db57600080fd5b50565b60805160601c60a051611
0f76101146000396000818161 04a4015261055501526000818161 01bc015261077b0152611
0f76000f3fe6080604052600436106100705760003560e01c80637fd6f15c1161004e5780637
fd6f15c146100f4578063bfb231d21461011f578063ca7dd375146101615780 63e067569814
61019e57610070565b806365e17c9d14610075578063696aae08146100a05780636bfb0d01
146100c9575b600080fd5b348015610081576 00080fd5b5061008a6101ba565b6040516100
979190610b59565b60405180910390f35b3480156100ac57600080fd5b506100c760048036
038101906100c29190610997565b6101de565b005b3480156100d557600080fd5b506100de
61049c565b6040516100eb9190610c82565b60405180910390f35b34801561010057600080
fd5b506101096104a2565b6040516101169190610c82565b60405180910390f35b34801561
012b57600080fd5b5061014660048036038101906101419190 6109fa565b6104c6565b6040
5161015896959493929190610ce2565b60405180910390f35b34801561016d57600080fd5b
5061018860048036038101906101839190 6109fa565b61054f565b6040516101959190610c
82565b60405180910390f35b6101b860048036038101906101b39190610a23565b6105b256
5b005b7f0000000000000000000000000000000000000000000000000000000000000000081
565b6002600054141561022457604051 7f08c379a000000000000000000000000000000000
000000000000000000000000815260040161021b90610c62565b60405180910390fd5b6002
6000819055506000821161026f576040517f08c379a0000000000000000000000000000000
0000000000000000000000000081526004016102669 0610c22565b60405180910390fd5b60
016000815480929190610282 90610ef9565b91905055508373ffffffffffffffffffffffffffffffffffff
ffff166323b872dd3330866040518463ffffffff1660e01b81526004016102c493929190610b74
565b600060405180830381600087803b1580156102de57600080fd5b505af11580156102f25
73d6000803e3d6000fd5b5050505060405180 60c0016040528060015481526020018573fffff
ffffffffffffffffffffffffffffffffff16815260200184815260200183815260200 13373ffffffffffffff

fffffffffffffffffffffff16815260200160001515815250600260006001548152602001908152602001600020600082015181600001556020820151816001016000610100a81548173fffffff
ffffffffffffffffffffffffffffffff02191690837 3fffffffffffffffffffffffffffffffffffffff1602179055506
040820151816002015560608201518160030155608082015181600401600061010 0a8154
8173ffffffffffffffffffffffffffffffffffffff02191690837 3fffffffffffffffffffffffffffffffffffffff1602
1790555060a08201518160040160146101000a81548160ff021916908315150217905550 90
50503373ffffffffffffffffffffffffffffffffffffff16836001547f68691bf1ff4da1d17e26d154d655
adee05f2c30e99638bfce346224ffd490995878686604051610486939291906 10bab565b6040
5180910390a4600160008190555050505050565b60015481565b7f0000000000000000000
0000000000000000000000000000000000000000081565b600260205280600052604 06
000206000915090508060000154908060010160009054906101000a900473ffffffffffffffffff
ffffffffffffffffff16908060020154908060030154908060040160009054906101000a90047
3ffffffffffffffffffffffffffffffffffffff16908060040160149054906101000a900460ff16905086
565b600060647f0000000000000000000000000000000000000000000000000000000000000000
0000606461058091906 10d54565b600260008581526020019081526020016000206003015
46105a19190610ddb565b6105ab9190610daa565b9050919050565b600260005414156105f
8576040517f08c379a0000000000000000000000000000000000000000000000000000000000
081526004016105ef90610c62565b60405180910390fd5b600260008190555060006 1060b8
361054f565b905060006002600085815260200190815260200160002090506000841180156
106365750600154841115 5b610675576040517f08c379a0000000000000000000000000000
0000000000000000000000000000081526004016106c90610c02565b60405180910390fd5
b813410156106b8576040517f08c379a0000000000000000000000000000000000000000000000
0000000000000081526004016106af90610c42565b60405180910390fd5b8060040160149
054906101000a900460ff161561070a576040517f08c379a0000000000000000000000000000
0000000000000000000000000081526004016070190610be2565b60405180910390fd
5b806004016000905490610100a900473ffffffffffffffffffffffffffffffffffffffff1673ffffffffffffff
ffffffffffffffffffffffffff166108fc8260030154908115029060405160006040518083818588
88f193505050501580156107 78573d6000803e3d6000fd5b507f0000000000000000000000
0000000000000000000000000000000000000000073ffffffffffffffffffffffffffffffffffffffff166
108fc8260030154846107c39190610e35565b90811502906040516000604051808303818588
888f193505050501580156107ee573d6000803e3d6000fd5b506001816004016014610100 0
a81548160ff02191690831515021790555080600101600090549061010 0a900473ffffffffff
ffffffffffffffffffffffffff1673ffffffffffffffffffffffffffffffffffffffff166323b872dd30338460020
1546040518463ffffffff1660e01b8152600401610871939291906 10b74565b6000604051808
30381600087803b15801561088b57600080fd5b505af115801561089f573d6000803e3d6000

fd5b505050503373ffffffffffffffffffffffffffffffffffff16816004016000905490610100 0a900473ffffffffffffffffffffffffffffffffffff1673ffffffffffffffffffffffffffffffffffffffff1682600201547fd78abc9e0d4dfe3f1e95dc2c5f8a6e6a3af57ab1174162e12eb01b661099ae00878560010160009054906101000a900473ffffffffffffffffffffffffffffffffffff16866003015489604051610957949392919061 0c9d565b60405180910390a4505060016000819055505050565b60008135905061097c81611093565b92915050565b60008135905061099816110aa565b92915050565b600080600080608085870312156109ad57600080fd5b60006109bb8782880161096d565b94505060206109cc8782880161098256 5b93505060406109dd8782880161098256 5b92505060606109ee8782880161098256 5b915050929591945092505650565b600060208284031215610a0c57600080fd5b6000610a1a84828501610982565b915050929150505650565b600080604083850312156103657600080fd5b6000610a448582860161098256 5b9250506020610a5585828601610982565b915050925092905056 5b610a6881610e7b565b82525050565b610a7781610e695565b82525050565b610a8681610e8d565b82525050565b610a9581610ed5565b82525050565b6000610aa8601183610d43565b9150610ab382610fa0565b602082019050919050565b6000610acb601383610d43565b9150610ad682610fc9565b602082019050919050565b60006010aee601f83610d43565b9150610af982610ff2565b602082019050919050565b6000610b11603783610d43565b9150610b1c8261101b565b604082019050919050565b6000610b34601f83610d43565b9150610b3f8261106a565b602082019050919050565b610b5381610ecb565b82525050565b60006020820190506 10b6e6000830184610a5f565b92915050565b6000606082019050610b896000830186610a6e565b610b966020830185610a6e565b610ba3604083018 4610b4a565b949350505050565b60006060820190506 10bc06000830186610a6e565b610bcd6020830185610b4a565b610bda6040830184610b4a565b949350505050565b600060208201905081810360008301526 10bfb81610a9b565b9050919050565b60006020820190508181036000830152610c1b81610abe565b9050919050565b60006020820190508181036000830152610c3b81610ae1565b9050919050565b60006020820190508181036000830152610c5b81610b04565b9050919050565b60006020820190508181036000830152610c7b81610b27565b9050919050565b60006020820190506 10c976000830184610b4a565b92915050565b60006080820190506 10cb26000830187610b4a565b610cbf6020830186610a6e565b610ccc6040830185610b4a565b610cd96060830184610b4a565b9594505050505050565b600060c082019050610cf76000830189610b4a565b610d046020830188610a8c565b610d1160408301876 10b4a565b610d1e6060830186610b4a565b610d2b6080830185610a5f565b610d3860a083018461 0a7d565b9796505050505050505050565b60008282526020820190509291505056 5b6000610d5f82610ecb565b9150610d6a83610ecb565b9250827fffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff03821115610d9f57610d9e610f42565b5b828201905092915050565b6000610db582610ecb565b9150610dc083610ecb565b925082610dd057610dcf610f71565b5b8282049

05092915050565b6000610de682610ecb565b9150610df183610ecb565b9250817fffffffffffff
ffffffffffffffffffffffffffffffffffffffffffffffffffffffff048311821515161561 0e2a57610e29610f42565
b5b8282029050929150505 65b6000610e4082610ecb565b9150610e4b83610ecb565b92508
2821015610e5e57610e5d610f42565b5b828203905092915050565b6000610e7482610eab5
65b9050919050565b6000610e8682610eab565b9050919050565b60008115159050919050 5
65b6000610ea482610e69565b9050919050565b600073ffffffffffffffffffffffffffffffffffffffff82
16905091905056 5b60008190509190505 65b6000610ee082610ee7565b9050919050565b60
00610ef282610eab565b9050919050565b6000610f0482610ecb565b91507fffffffffffffffffffff
ffffffffffffffffffffffffffffffffffffffffffff821415610f3757610f36610f42565b5b6001820190509
19050565b7f4e487b7100000000000000000000000000000000000000000000000000000000
0600052601160045260246000fd5b7f4e487b7100000000000000000000000000000000000000
0000000000000000000000600052601260045260246000fd5b7f6974656d20616c726561647
920736f6c6400000000000000000000000000000000600082015250565b7f6974656d20646f6
5736e2774206578697374730000000000000000000000000600082015250565b7f5072696
365206d7573742062652067726561746572207468616e207a65726f00600082015250565b7
f6e6f7420656e6f756768206574686572207 46f20636f766572207468652069746000820152
7f656d2070726963520616e64206d61726b65742066656500000000000000000060208201
5250565b7f5265656e7472616e637947561726 43a207265656e7472616e742063616c6c00
600082015250565b61109c81610e99565b81146110a757600080fd5b50565b6110b381610e
cb565b81146110be57600080fd5b5056fea2646970667358221220ef8cb3e7c48eeed4dd8738
060498668254ac9ba29367f47aa3d37adc493d08e764736f6c63430008040033

Similarly, compiling with abi flag returns the JSON deployed object, with which the Frontend application interacts.

solc Marketplace.sol –abi

{
"_format": "hh-sol-artifact-1",
"contractName": "Marketplace",
"sourceName": "src/backend/contracts/Marketplace.sol",
"abi": [
{

```
  "inputs": [
    {
      "internalType": "uint256",
      "name": "_feePercent",
      "type": "uint256"
    }
  ],
  "stateMutability": "nonpayable",
  "type": "constructor"
},
{
  "anonymous": false,
  "inputs": [
    {
      "indexed": false,
      "internalType": "uint256",
      "name": "itemId",
      "type": "uint256"
    },
    {
      "indexed": false,
      "internalType": "address",
      "name": "nft",
      "type": "address"
    },
    {
      "indexed": true,
      "internalType": "uint256",
      "name": "tokenId",
      "type": "uint256"
    },
    {
      "indexed": false,
      "internalType": "uint256",
```

```
      "name": "price",
      "type": "uint256"
    },
    {
      "indexed": true,
      "internalType": "address",
      "name": "seller",
      "type": "address"
    },
    {
      "indexed": true,
      "internalType": "address",
      "name": "buyer",
      "type": "address"
    },
    {
      "indexed": false,
      "internalType": "uint256",
      "name": "boughtDateTime",
      "type": "uint256"
    }
  ],
  "name": "Bought",
  "type": "event"
},
{
  "anonymous": false,
  "inputs": [
    {
      "indexed": true,
      "internalType": "uint256",
      "name": "itemId",
      "type": "uint256"
    },
```

```
    {
      "indexed": false,
      "internalType": "address",
      "name": "nft",
      "type": "address"
    },
    {
      "indexed": true,
      "internalType": "uint256",
      "name": "tokenId",
      "type": "uint256"
    },
    {
      "indexed": false,
      "internalType": "uint256",
      "name": "price",
      "type": "uint256"
    },
    {
      "indexed": true,
      "internalType": "address",
      "name": "seller",
      "type": "address"
    },
    {
      "indexed": false,
      "internalType": "uint256",
      "name": "offeredDateTime",
      "type": "uint256"
    }
  ],
  "name": "Offered",
  "type": "event"
},
```

```
{
 "inputs": [],
 "name": "feeAccount",
 "outputs": [
  {
   "internalType": "address payable",
   "name": "",
   "type": "address"
  }
 ],
 "stateMutability": "view",
 "type": "function"
},
{
 "inputs": [],
 "name": "feePercent",
 "outputs": [
  {
   "internalType": "uint256",
   "name": "",
   "type": "uint256"
  }
 ],
 "stateMutability": "view",
 "type": "function"
},
{
 "inputs": [
  {
   "internalType": "uint256",
   "name": "_itemId",
   "type": "uint256"
  }
 ],
```

```
  "name": "getTotalPrice",
  "outputs": [
    {
      "internalType": "uint256",
      "name": "",
      "type": "uint256"
    }
  ],
  "stateMutability": "view",
  "type": "function"
},
{
  "inputs": [],
  "name": "itemCount",
  "outputs": [
    {
      "internalType": "uint256",
      "name": "",
      "type": "uint256"
    }
  ],
  "stateMutability": "view",
  "type": "function"
},
{
  "inputs": [
    {
      "internalType": "uint256",
      "name": "",
      "type": "uint256"
    }
  ],
  "name": "items",
  "outputs": [
```

```
    {
      "internalType": "uint256",
      "name": "itemId",
      "type": "uint256"
    },
    {
      "internalType": "contract IERC721",
      "name": "nft",
      "type": "address"
    },
    {
      "internalType": "uint256",
      "name": "tokenId",
      "type": "uint256"
    },
    {
      "internalType": "uint256",
      "name": "price",
      "type": "uint256"
    },
    {
      "internalType": "address payable",
      "name": "seller",
      "type": "address"
    },
    {
      "internalType": "bool",
      "name": "sold",
      "type": "bool"
    }
  ],
  "stateMutability": "view",
  "type": "function"
},
```

```
{
  "inputs": [
    {
      "internalType": "contract IERC721",
      "name": "_nft",
      "type": "address"
    },
    {
      "internalType": "uint256",
      "name": "_tokenId",
      "type": "uint256"
    },
    {
      "internalType": "uint256",
      "name": "_price",
      "type": "uint256"
    },
    {
      "internalType": "uint256",
      "name": "offeredDateTime",
      "type": "uint256"
    }
  ],
  "name": "makeItem",
  "outputs": [],
  "stateMutability": "nonpayable",
  "type": "function"
},
{
  "inputs": [
    {
      "internalType": "uint256",
      "name": "_itemId",
      "type": "uint256"
```

```
      },
      {
        "internalType": "uint256",
        "name": "boughtDateTime",
        "type": "uint256"
      }
    ],
    "name": "purchaseItem",
    "outputs": [],
    "stateMutability": "payable",
    "type": "function"
  }
],
```

2. **Setup the blockchain network or use the public blockchain network**

Local blockchain networks can be set up with Ganache or Hardhat which provides the private key along with some ETH balance on it.

npx hardhat node

3. **Write the deployment script**

Deployment script is the javascript function that actually connects to the blockchain network and sends transactions on it.

```
async function deploy(){
        let r = await web3.eth.sendTransaction({
        from: myAccount.address,
        gas: 8000,
        data: bytecode    })
}
```

4. **Store the Contract address**

Once a smart contract is deployed, it returns the contract address along with the block number. This is how smart contracts are deployed in the Ethereum blockchain network.

Chapter 5: Implementation and Testing

5.1 Implementation

Implementation of the system was carried out after the system was designed. The front-end and back-end logic of the application were both catered to provide maximum client satisfaction and fulfillment of the objectives that were initially defined in the project.

5.1.1 Tools and Technologies

1. The front-end of the system was developed using React and Javascript. The reason to choose these frameworks is that it is easy to learn, flexible, and provide speed and performance.
2. The backend of the project is supported by the development of smart contracts that are written using the Solidity programming language. And Hardhat framework is used to deploy the smart contract into the Ethereum network.
3. Product is stored in a Decentralized network Blockchain. So Blockchain and Ethereum networks act as a decentralized database for us.
4. Project Libre was used for project management.
5. Github was used for version control and collaboration among the team members.

5.1.2 Implementation Details of Modules

The implementation of modules can be differentiated into major categories that define the functionalities of the application and objectives defined for this project work. Some of those categories include Wallet connection, adding product to the blockchain network, purchasing the product, and resulting from the product. Currently, some of the modules have yet not been finished such as storing the buyer and seller's basic information in a relational database.

5.1.2.1 Wallet Connection with Metamask

As users have to connect their wallet address using a browser extension such as Metamask.

Method for wallet connection with metamask

```
const web3Handler = async () => {
  const accounts = await window.ethereum.request({ method: 'eth_requestAccounts'
})
  setAccount(accounts[0])
  const provider = new ethers.providers.Web3Provider(window.ethereum)
  const signer = provider.getSigner()
  loadContracts(signer)
 }
```

5.1.1.1 Add product for sale

This system allows the seller to add products for sale. Currently, the system supports a limited number of product categories. To add a product for sale, the seller has to complete three steps of verification along with a certain gas fee. The three steps are NFT mint, approval for the marketplace, and contract verification.

Method to add product

```
const createProduct = async () =>
 {
if (!image || !price || !name || !description) return;
 try {
const result = await client.add( JSON.stringify(
{ image, price, name, description, productType, vehicleNo, lotNo,
deviceIdentificationNumber, propertyVerificationNumber, purchaseDate }
) );
 mintThenList(result); } catch (error)
{ console.log("ipfs uri upload error: ", error); } };
```

5.1.1.2 Purchase product

The system allows buyers to purchase the product from hosted Ethereum network. The buyer needs to pay the amount of the product along with the 1 percent market price.

Method to add product

```
const purchases = await Promise.all(results.map(async i =>
{
i = i.args
const uri = await product.tokenURI(i.tokenId) // use uri to fetch the product metadata stored on ipfs
const response = await fetch(uri)
const metadata = await response.json() // get total price of item (item price + fee)
const totalPrice = await marketplace.getTotalPrice(i.itemId) // define listed item object
let purchasedItem = { totalPrice,
 price: i.price,
 itemId: i.itemId,
 name: metadata.name,
 description: metadata.description,
 image: metadata.image }
 return purchasedItem }))
```

5.1.1.3 Resale the product

The system allows buyers to re-sale the product they have purchased through the blockchain network. During reselling, they can edit the project name, description, and the price of the product.

Method to resale Product

```
const resaleProduct = async () => {
    const image = items[0].image;
```

```
    const price = items[0].price;
    const lotNo = items[0].lotNo;
    const vehicleNo = items[0].vehicleNo;
    const propertyVerificationNumber = items[0].propertyVerificationNumber;

    const deviceIdentificationNumber = items[0].deviceIdentificationNumber
    const productType = items[0].productType;
    const nft = items[0].nft;
    try {
      const result = await client.add(
        JSON.stringify({ image,
 price,
 name,
description,
productType,
vehicleNo,
lotNo,
deviceIdentificationNumber,
propertyVerificationNumber })
      );
      mintThenList(result, nft);
    } catch (error) {
      console.log("ipfs uri upload error: ", error);
    }
  };
```

5.2 Testing

5.2.1 Test Cases for Unit Testing

Table 1: Test Cases for Unit Testing and Result

S.N.	Test Case	Expected Outcome	Result
Wallet Connection			

1.	User tries to connect their Wallet	The user wallet is connected	SUCCESS
Add Product for Sale			
2	The user clicks on the Dashboard	The user is directed to the dashboard page	SUCCESS
3	In the dashboard, the user clicks sale the product	The user is directed to the form, where they can add product	SUCCESS
4	User select the product type and enter the detail based on the product type selected	The user is prompt with additional fields	SUCCESS
5	If the user did not fill all the Information	The form is not submitted and the user is greeted with an error message.	SUCCESS
6	In the add product form, the user enters valid data and presses submit button.	The form is successfully submitted and data is stored in the blockchain network.	SUCCESS
Buyer Purchase Product			
7	User clicks on the Products Menu.	The user is directed to the products listing page.	SUCCESS
8	The user clicks on the buy now Button	The user is prompted with metamask wallet to pay the amount of product.	SUCCESS
9	The user clicks on the product Card	The user is directed to the product detail page	SUCCESS
10	The user clicks the buy now a button from the product detail page	The user is prompted with a meta maskwallet to pay the amount of the product.	SUCCESS
Resale the product			
11	The user clicks the dashboard Menu	The user is directed to the user dashboard.	SUCCESS

12	A user clicks on the purchased List	The user is directed to specific pages.	SUCCESS
13	A user clicks on a resale product	The meta mask wallet is prompted andcompletes the steps of NFT mint, approval for the marketplace, and contract sign	SUCCESS

5.2.2 System Testing

During the development of the system, it was found that due to the use of loops used in the

system programming, the bytecode of the smart contract turned a bit larger than expected. This means that it requires more gas to be executed i.e. that failure might occur if the gas limit is exceeded.[5]

5.3 Result Analysis

For 13 different unit test cases under four major functionalities, the application so far has remained successful. This is a positive result as the success of these functions fulfills the objectives of the project.

Chapter 6: Conclusion and Future Recommendations

6.1. Conclusion

After the completion of the project, a decentralized, reliable, secure, and third-party independent marketplace for buying and selling products was developed. With the creation of Smart Contract, deployed successfully in the Ropsten test network, a full functioning DApp using Truffle was made.

6.2. Future Recommendations

Limitations and future works
Limitations

2. This system is only applicable to larger organizations.

3. Since the system is running in Ethereum, a private blockchain should be set up touse for the marketplace.

4. Less knowledge of people about blockchain technology.Future enhancements

1. Make revolutionary changes at the national level.

2. Create a digital identity using Uport.

References

[1] A Blockchain-Based Application System for Product Anti-Counterfeiting (JINHUA MA, SHIH-YA LIN, XIN CHEN , HUNG-MIN SUN , YEH-CHENG CHEN , (Grad-uate Student Member, IEEE) AND HUAXIONG WANG) https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8985337

[2] Counterfeited Product Identification in a Supply Chain using Blockchain Technology https://www.researchgate.net/publication/353971876_Counterfeited_Product_Identification_in_a_Supply_Chain_using_Blockchain_Technology

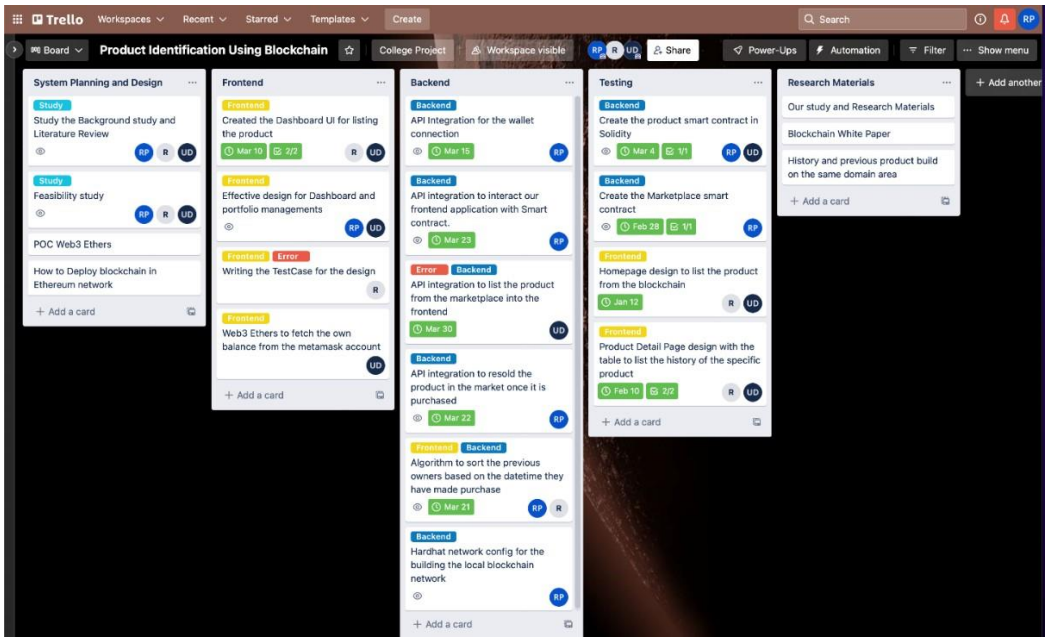
[3] Enterprise blockchain for verifying product authenticity (by Radhika Iyengar and Jor-den Woods) https://shorturl.at/cdlBZ

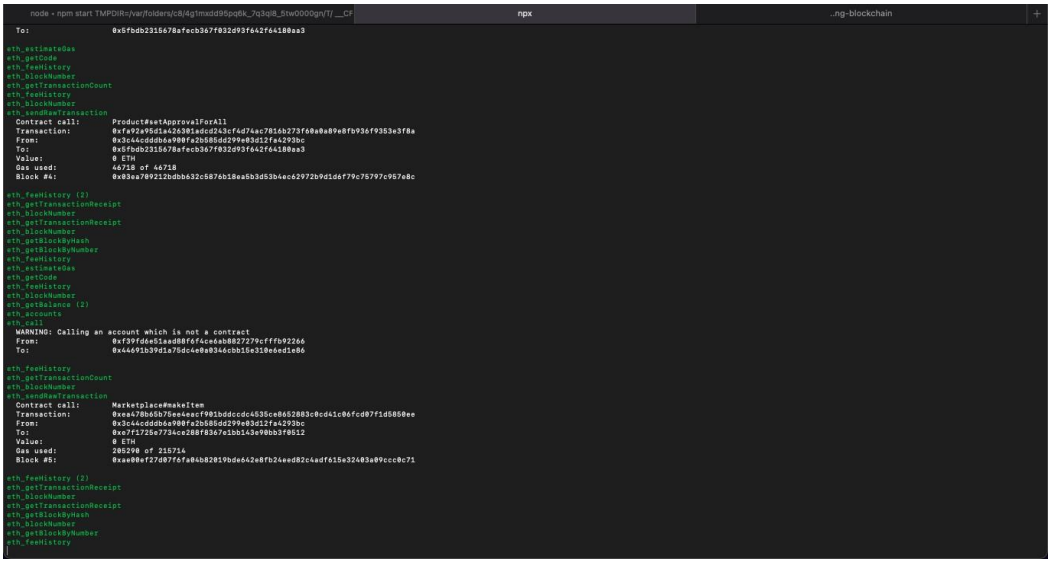
[4] "ERC721," *OpenZeppelin Docs*. [Online]. Available: https://docs.openzeppelin.com/contracts/3.x/erc721.

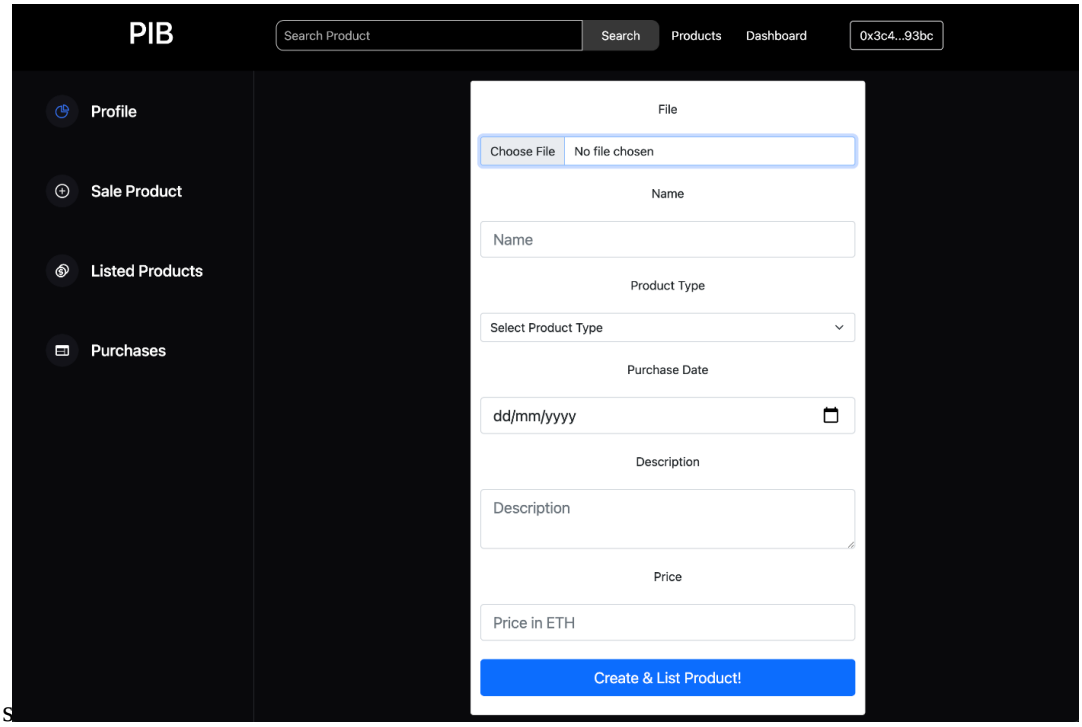
[5] W. Schwab, "Everything you ever wanted to know about events and logs on ethereum," *Medium*, 08-Sep-2020. [Online]. Available: https://medium.com/linum-labs/everything- you-ever-wanted-to-know-about-events-and-logs-on-ethereum-fec84ea7d0a5.

Appendix

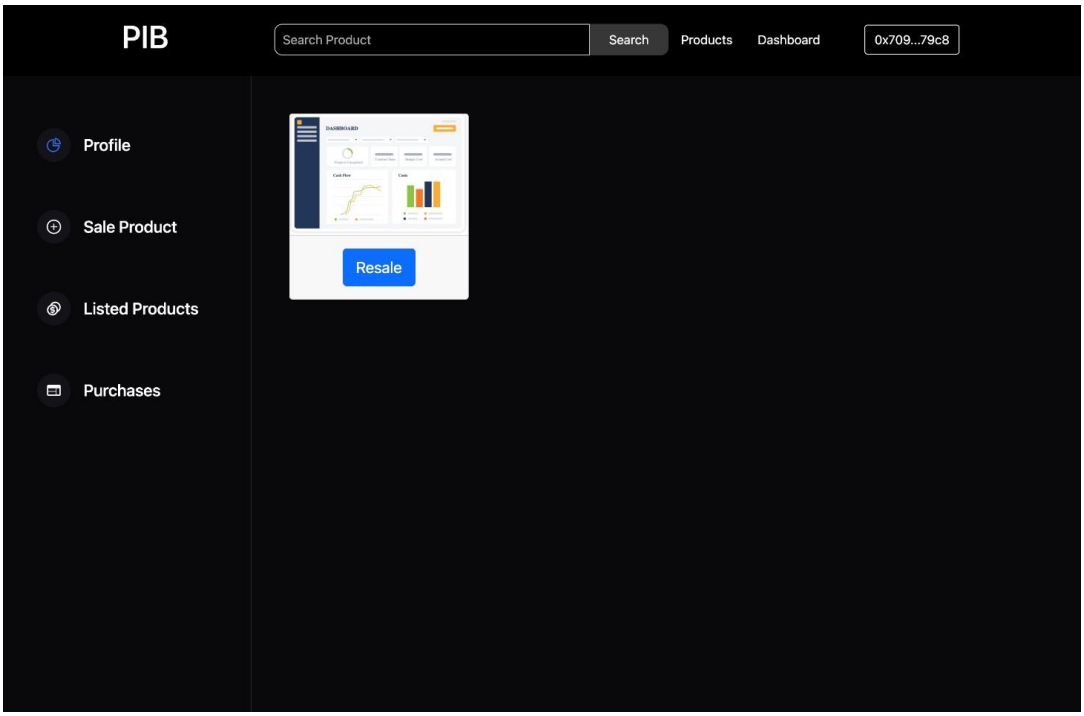
Trello




node • npm start TMPDIR=/var/folders/c8/4g1mxdd95pq6k_7q3qj8_5tw0000gn/T/ __CF
npx
..ng-blockchain
To: 0x5fbdb2315678afecb367f032d93f642f64180aa3
eth_estimateGas
eth_getCode
eth_feeHistory
eth_blockNumber
eth_getTransactionCount
eth_feeHistory
eth_blockNumber
eth_sendRawTransaction
Contract call: Product#setApprovalForAll
Transaction: 0xfa92a95d1a426381a6cd243cf4d74ac701bb273fe0a0a89e8fb936f9353e3f8a
From: 0x3c44cdddb6a900fa2b585dd299e03d12fa4293bc
To: 0x5fbdb2315678afecb367f032d93f642f64180aa3
Value: 0 ETH
Gas used: 46718 of 46718
Block #4: 0x03ea709212bdbb632c5876b18ea5b3d53b4ec62972b9d1d6f79c75797c957e8c
eth_feeHistory (2)
eth_getTransactionReceipt
eth_blockNumber
eth_getTransactionReceipt
eth_blockNumber
eth_getBlockByHash
eth_getBlockByNumber
eth_feeHistory
eth_estimateGas
eth_getCode
eth_feeHistory
eth_blockNumber
eth_getBalance (2)
eth_accounts
eth_call
WARNING: Calling an account which is not a contract
From: 0xf39fd6e51aad88f6f4ce6ab8827279cfffb92266
To: 0x44691b39d1a75dc4e0a0346cbb15e310e6ed1e86
eth_feeHistory
eth_getTransactionCount
eth_blockNumber
eth_sendRawTransaction
Contract call: Marketplace#makeItem
Transaction: 0xea478b65b75ee4eacf981bddccdc4535ce8652883c0cd41c06fcd07f1d5850ee
From: 0x3c44cdddb6a900fa2b585dd299e03d12fa4293bc
To: 0xe7f1725e7734ce288f8367e1bb143e90bb3f0512
Value: 0 ETH
Gas used: 285298 of 215714
Block #5: 0xae80ef27d07f6fa04b82019bde642e8fb24eed82c4adf615e32403a09ccc0c71
eth_feeHistory (2)
eth_getTransactionReceipt
eth_blockNumber
eth_getTransactionReceipt
eth_getBlockByHash
eth_blockNumber
eth_getBlockByNumber
eth_feeHistory


PIB
Search Product
Search
Products
Dashboard
0x3c4...93bc
Profile
Sale Product
Listed Products
Purchases
File
Choose File
No file chosen
Name
Name
Product Type
Select Product Type
Purchase Date
dd/mm/yyyy
Description
Description
Price
Price in ETH
Create & List Product!

S


PIB
Search Product
Search
Products
Dashboard
0x709...79c8
Profile
Sale Product
Listed Products
Purchases
Resale