

A Project Report
On
**Food Supply Chain Management Using Smart Contracts in Blockchain
Technology**

Submitted in Partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

SUBMITTED

By

ASHWINI TARUN - 19671A1207

THUMAR JANKIBEN NARSHIBHAI - 20675A1206

Under the esteemed guidance of

M. A. MUNEER

ASSISTANT PROFESSOR



Department of Information Technology

Accredited by NBA

J.B. Institute of Engineering & Technology

(UGC AUTONOMOUS)

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad)

Baskar Nagar, Yenkapally, Moinabad Mandal, R.R. District, Telangana (India)-500075

2022-2023

J.B. INSTITUTE OF ENGINEERING & TECHNOLOGY

(UGC AUTONOMOUS)

(Accredited by NAAC, Permanently Affiliated to JNTUH)

Bhaskar Nagar, Yenkapally, Moinabad Mandal, R.R. Dist.Telangana(India) -500 075

DEPARTMENT OF INFORMATION TECHNOLOGY



Accredited by NBA

CERTIFICATE

This is to certify that the project report entitled “**Food Supply Chain Management Using Smart Contracts in Blockchain Technology**” being submitted to the Department of Information Technology, J.B. Institute of Engineering and Technology, in accordance with Jawaharlal Nehru Technological University regulations as partial fulfillment required for successful completion of Bachelor of Technology is a record of bonafide work carried out during the academic year 2022-23 by,

ASHWINI TARUN - 19671A1207

THUMAR JANKIBEN NARSHIBHAI - 20675A1206

Internal Guide

M. A. MUNEEER
ASSISTANT PROFESSOR

Head of the Department

Dr. L. SRIDHARA RAO
ASSOCIATE PROFESSOR

External Examiner

J.B. INSTITUTE OF ENGINEERING & TECHNOLOGY

(UGC AUTONOMOUS)

(Accredited by NAAC, Permanently Affiliated to JNTUH)

Baskar Nagar, Yenkapally, Moinabad Mandal, R.R. Dist.Telangana(India) -500 075

DEPARTMENT OF INFORMATION TECHNOLOGY

Accredited by NBA



DECLARATION

We hereby certify that the Main Project report entitled “**Food Supply Chain Management Using Smart Contracts in Blockchain Technology**” carried out under the guidance of **M. A. MUNEER** , **Assistant Professor** is submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Information Technology**. This is a record of bonafide work carried out by us and the results embodied in this project report have not been reproduced or copied from any source. The results embodied in this project report have not been submitted to any other university or institute for the award of any other degree or diploma.

Date:

Place:

ASHWINI TARUN – 19671A1207

THUMAR JANKIBEN NARSHIBHAI – 20675A1206

ACKNOWLEDGEMENT

At outset we express our gratitude to almighty lord for showering his grace and blessings upon us to complete this Main Project. Although our name appears on the cover of this book, many people had contributed in some form or the other to this project Development. We could not have done this Project without the assistance or support of each of the following.

First of all we are highly indebted to **Dr. P. C. KRISHNAMACHARY**, Principal for giving us the permission to carry out this Main Project.

We would like to thank **Dr. L. SRIDHARA RAO**, Associate Professor & Head of the Department of INFORMATION TECHNOLOGY, for being moral support throughout the period of the study in the Department.

We are grateful to **M. A. MUNEER**, Assistant Professor of the Department of INFORMATION TECHNOLOGY, for his valuable suggestions and guidance given by him during the execution of this Project work.

We would like to thank Teaching and Non-Teaching Staff of Department of Information Technology for sharing their knowledge with us.

ASHWINI TARUN - 19671A1207

THUMAR JANKIBEN NARSHBHAI – 20675A1206



J. B. INSTITUTE OF ENGINEERING & TECHNOLOGY

(UGC AUTONOMOUS)

Accredited by NAAC & NBA, Approved by AICTE, Permanently Affiliated to JNTUH
HYDERABAD



CERTIFICATE OF PRESENTATION

Cert No :-
JBIETAAATM10032023AUTP0111



This is to certify Dr. / Mr. / Ms. / Mrs.

Tarun Ashwini

For presenting at J BHASKAR RAO MEMORIAL 1ST INTERNATIONAL
CONFERENCE ON

"ADVANCED APPLICATIONS OF TECHNOLOGY AND MANAGEMENT"
(AATM-2023) FROM 10TH-11TH MARCH 2023.

DR. NIRAJ UPADHAYAYA,
PROGRAM CHAIR

DR. TOWHEED SULTANA
PROGRAM CHAIR

DR. P.C KRISHNAMACHARY
PRINCIPAL, JBIET

P.C. Krishnamachary

Towheed's



J. B. INSTITUTE OF ENGINEERING & TECHNOLOGY

(UGC AUTONOMOUS)

Accredited by NAAC & NBA, Approved by AICTE, Permanently Affiliated to JNTUH
HYDERABAD



CERTIFICATE OF PRESENTATION

Cert No :-
JBIETAAATM10032023AUTP0112



This is to certify Dr. / Mr. / Ms. / Mrs.

Thumar Jankiben Narshibhai

For presenting at J BHASKAR RAO MEMORIAL 1ST INTERNATIONAL

CONFERENCE ON

"ADVANCED APPLICATIONS OF TECHNOLOGY AND MANAGEMENT"

(AATM-2023) FROM 10TH-11TH MARCH 2023.

[Signature]

DR. NIRAJ UPADHAYAYA,
PROGRAM CHAIR

[Signature]

DR. TOWHEED SULTANA
PROGRAM CHAIR

[Signature]

DR. P.C KRISHNAMACHARY
PRINCIPAL, JBIET



J. B. INSTITUTE OF ENGINEERING & TECHNOLOGY

UJGS AUTONOMOUS

Accredited by NAAC & NBA, Approved by AICTE, Permanently Affiliated to JNTUH
HYDERABAD



CERTIFICATE OF PRESENTATION

Cert No :-
JBIEATAATM10032023AUTP0110



This is to certify **Dr. / Mr. / Ms. / Mrs.**

Muneer M A

For presenting at J BHASKAR RAO MEMORIAL 1ST INTERNATIONAL

CONFERENCE ON

"ADVANCED APPLICATIONS OF TECHNOLOGY AND MANAGEMENT"

(AATM-2023) FROM 10TH-11TH MARCH 2023.

DR. NIRAJ UPADHAYAYA,
PROGRAM CHAIR

DR. TOWHEED SULTANA
PROGRAM CHAIR

DR. P.C KRISHNAMACHARY
PRINCIPAL, JBIE

P.C. Krishnamachary

Towheed's

Niraj Upadhyaya

ABSTRACT

The management of the food supply chain is essential for moving food items from farmers to consumers. The functioning of the food supply chain management has undergone a significant amount of innovation as a result of the quick development of contemporary internet technologies, in order to fulfil consumer demand for food goods and provide an effective product to the general public. A central authority controls and maintains everything, which causes issues like information on the supply of rice and its procedures being tampered with and falsified. Blockchain is a groundbreaking technology that is decentralised and distributed, not controlled by a single entity, and it offers distributed databases to provide transparency in the supply chain network.

It is suggested to test a supply chain simulation for the food products sector. We first determine the prerequisites for these reference datasets, and then we establish the fundamental components. A simulation model from the measurement and improvement of food product production is used to build the supply chain nodes that reflect the farmer who cultivates the crops.

CONTENTS

| SL. No | Name of the Topic | Page No |
|---------------|--|----------------|
| | Certificate | I |
| | Declaration | II |
| | Acknowledgement | III |
| | Abstract | IV |
| 1. | INTRODUCTION | 1 |
| 2. | LITERATURE SURVEY | 3 |
| 3. | SYSTEM ANALYSIS | |
| | 3.1 Existing System | 5 |
| | 3.2 Drawbacks of Existing System | 5 |
| | 3.3 Proposed System | 6 |
| | 3.4 Advantages of Proposed System | 6 |
| | 3.5 Software Development Life Cycle Model | 6 |
| | 3.6 Project Implementation Plan | 9 |
| 4. | SOFTWARE REQUIREMENT SPECIFICATIONS | |
| | 4.1 Functional Requirements | 10 |
| | 4.2 Non-Functional Requirements | 10 |
| | 4.3 Software Requirement Specifications | 11 |
| | 4.4 Hardware Requirement Specifications | 11 |
| 5. | SYSTEM DESIGN | |
| | 5.1 System Architecture. | 12 |
| | 5.2 Design Tool Used | 13 |
| | 5.3 UML Diagrams | |
| | 5.3.1 Use Case Diagram | 16 |
| | 5.3.2 Class Diagram | 17 |
| | 5.3.3 Sequence Diagram | 18 |

| | | |
|------------|------------------------------------|----|
| 5.3.4 | Activity Diagram | 19 |
| 6. | IMPLEMENTATION | |
| 6.1 | Introduction | 20 |
| 6.2 | Technology Used | 24 |
| 6.3 | Coding Standards | 26 |
| 6.4 | Modules | 29 |
| 7. | SYSTEM TESTING | |
| 7.1 | Test Plan | 32 |
| 7.2 | Software Testing | 33 |
| 7.3 | Test Cases | 37 |
| 7.4 | Bug Report | 38 |
| 8. | RESULT SCREENS | 39 |
| 9. | CONCLUSION AND FUTURE SCOPE | 43 |
| 10. | BIBLIOGRAPHIES | 44 |
| | References | |
| 11. | APPENDIXES | 46 |
| 11.1 | Sample Code | |

LIST OF FIGURES

| SL .No | Description of Figure | Page No |
|--------|------------------------------------|---------|
| 1 | Spiral Model | 8 |
| 2 | Project Implementation Plan | 9 |
| 3 | System Architecture | 12 |
| 4 | Interface of Visual Paradigm | 15 |
| 5 | Use Case Diagram | 16 |
| 6 | Class Diagram | 17 |
| 7 | Sequence Diagram | 18 |
| 8 | Activity Diagram | 19 |
| 9 | Ethereum Virtual Machine | 22 |
| 10 | DMAIC | 27 |
| 11 | Testing Plan | 32 |
| 12 | Product Registration Interface - 1 | 39 |
| 13 | Quality Testing Interface - 2 | 40 |
| 14 | Customer Interface - 3 | 41 |
| 15 | Micro-Finance Interface – 4 | 42 |

LIST OF TABLES

| SL. No. | Description of Tables | Page No. |
|--------------------|------------------------------|---------------------|
| 1 | Unit Test Cases | 37 |
| 2 | Bug Report - 1 | 38 |
| 3 | Bug Report - 2 | 38 |
| 4 | Bug Report - 3 | 38 |

INTRODUCTION

1. INTRODUCTION

The Supply Chain Management (SCM) is a group of processes and sub-processes carried out for transforming raw material into a final product, maximizing customer value and achieving a maintainable competitive advantage. It is also interpreted as a network of entities that are part of the system from production to trading. The whole supply chain network is divided into several stages. Processes involved in these stages often take months to complete. In such situation, if the final product lacks in quality, it becomes extremely difficult to track the root cause of the problem. The demand for top quality products and interest of end consumers in the provenance of data is increasing rapidly. Therefore, it has become necessary for every supply chain system to track the movement of products from origin to the end consumers.

To gain end consumers' trust, the supply chain authorities have to be efficient and accurate in delivering information. Agriculture development is predicated by improvement in farm production and productivity, better utilization of agriculture inputs, proper marketing infrastructure and support, and also efficient food management. But currently traditional agriculture supply chain management facing many problems in terms of centralized network, lack of trust, less quality product and lack of communication. By Introducing blockchain in traditional agriculture supply chain will overcome the problem that it is facing today. Blockchain is secure system that plays a significant role in evolution of supply chain with its inherent properties like decentralization, transparency and immutability.

Agriculture is very essential for the living of the majority of population in TamilNadu. In terms of productivity of important and main crops, Tamil Nadu is in the lead. Achieving food security by increasing agricultural production forms the core of agricultural development strategy of the State. Agriculture development is predicated by improvement in farm production and productivity, better utilization of agriculture inputs, proper marketing infrastructure and support, and also efficient food management. Food safety seems to be an important area where there is an interest from

both the producers and also the consumers. In Supply Chains, [7] by bringing transparency it helps us in improving the processes involved in production. Traceability is also very important that we can know the source of the item including details like the producer details, harvested and produced time etc., Agriculture system in Tamil Nadu requires a huge improvement to meet the increasing demands of it. Advanced technologies are being developed to meet the complex agricultural challenges faced by Tamil Nadu. Among those technologies, Blockchain is the most emerging technology. It is based on cryptographic hash. It is a decentralized and encrypted ledger system for storing transactions. This ensures that the transactions and the identity of the user can never be compromised.

If such a fraudulent transactions occurs, the decentralized mining system will block it to enter into the encrypted chain. The most obvious application of Blockchain Technology is the supply chain logistics. In logistics, blockchain gives a lot of options related to shipment data. All products and items could be tracked and this can help in preparing for any expected delays due to any conditions. We also have applications of Blockchain Technology (BCT) where assets are traded in exchange of money. In Agriculture Supply Chain Management, the transaction may include the data like quantity, raw materials, etc., Several crop insurance schemes like the National Agricultural Insurance Scheme, can be maintained using Blockchain Technology for tamper proof records and for periodic checking of settlement of claims during crop losses. The main advantage of Blockchain Technology is that it eliminates the need for third-party representatives as smart contracts can be used to settle transactions.

LITERATURE SURVEY

2. LITERATURE SURVEY

2.1 Definition of Terms:

In this day and age, agriculture producers are faced with multiple obstacle, from seasonal changes to the broken supply chain; their occupation is very laborious and demanding. In such situation , a distinct information database consisting of credible information would be very helpful. Transfer of knowledge is important in all aspects may cause the spread of misinformation. This can be curbed by using blockchain, a data ledger which is reliable and incorruptible. Here an attempt to analyse the different ways in which blockchain technology can be incorporated in the agricultural supply chain, as a transparent and dependable transaction mechanism is explored.

Satoshi Nakamoto in his paper “Bitcoin: A Peer-to-Peer Electronic Cash System”, has explained how this digital currency which was built on Blockchain platform could challenge the existing financial systems and eradicate the problems related to high transaction fees, global transactions, double spending, inflation etc [1]. Blockchain technology is not limited only to Bitcoin as there are various applications which can be built upon this framework. Blockchain groups all the digital transactions into a block and all the blocks are arranged in chronological order thus forming a chain of blocks which is Blockchain [2]. Blockchain uses some cryptographic functions to encrypt the data so that anonymity of a person or data is maintained throughout the Blockchain network [3]. There are different types of Blockchain such as Public, Private and Consortium Blockchain based on necessity of an organization or individual want to use[4]. A Public Blockchain is completely decentralized allowing its connected users to read or write data onto the Blockchain.

Whereas in Private Blockchain, permissions to read or write data onto the Blockchain are controlled by one organization which is highly trusted by other users. Consortium Blockchain is hybrid of public and private Blockchain [5]. Instead of allowing any person

to participate in validation process Consortium Blockchain here allows group of organizations to form together to have full control over Blockchain and users ought to participate are predetermined [6]. Hence, depending on the necessity and type of data to be uploaded anyone of these Blockchain models is to be used.

agricultural SCM System facing many obstacles. This paper explains how blockchain technology helpful for SCM to transfer the information in secured way. The Third-party interference in this aspect curbed using a data ledger which is reliable and incorruptible [7]. It analyse the different ways in which blockchain technology can be incorporated in the agricultural supply chain, as a transparent and depend able transaction mechanism. A fully decentralized blockchain based traceability that enables to build blocks for agriculture that continuously integrate with IoT devices from provider to consumer [8]. To implement, we introduced Provider-Consumer Network a theoretical end to end food traceability application.

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1. EXISTING SYSTEM

The Existing system is a Traditional Agricultural Supply Chain Management. The traditional Agricultural Supply chain is highly fragmented. It consists of multiple agents or intermediaries tapping into the marketing channel to realize a profit and successfully pass on the losses to the producer. the farmer cultivates the crop and sell the food product to the intermediaries at a margin price. The food products are reached to end consumer by passing through multiple intermediaries at increasing cost. The farmer sell the products at the lower cost but it reaches to the end customer at higher price where only intermediaries are yields more profit than the farmer who harvest it. They may also hoard the products and create a artificial demand so that they can sell the products at the higher price.

3.2 Drawbacks of Existing System

- Both expensive and time-consuming.
- There is a chances of occurrence of Food Fraud.
- It very difficult to trace the product.
- Lack of traceability may occur for Illegal Production.
- Unable to handle large amount of data.
- Maintaining manual records are vulnerable to inaccurate updating and my consumes much manual power.

3.3 PROPOSED SYSTEM

In this paper we propose a method through which the number plate recognition can be achieved more precisely. Here we use K-Nearest Neighbor technique for character recognition. The first stage is capturing the license plate images for dataset collection.

These images will be

used as the training and testing data. Second, the image pre-processing stage includes cropping, resizing and Gray scaling. The third stage is the segmentation process includes edge detection and thresholding. The fourth stage is feature extraction which applies contour detection. In the next stage, all the images from the previous stage are segmented to obtain letters and numbers images. These images will be used as the input for training and testing data. The training and testing data are processed by using K-Nearest Neighbor (KNN) algorithm.

3.4 Advantages of Proposed System

- More efficient than existing approaches.
- It may help classifying the things very fast.
- It may produce high accuracy.
- It is much more feasible and effective in classifying the elements in the segmentation phases.

3.5 Software Development Life Cycle Model

SDLC METHDOLOGIES





This document play a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system. It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process. SPIRAL MODEL was defined by Barry Boehm in his 1988 article, “A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration models.

As originally envisioned, the iterations were typically 6 months to 2 years long. Each phase starts with a design goal and ends with a client reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project.

The steps for Spiral Model can be generalized as follows:

- The new system requirements are defined in as much details as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
- A preliminary design is created for the new system.
- A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.

A second prototype is evolved by a fourfold procedure:

-  Evaluating the first prototype in terms of its strengths, weakness, and risks.
-  Defining the requirements of the second prototype.
-  Planning and designing the second prototype.
-  Constructing and testing the second prototype.
- At the customer option, the entire project can be aborted if the risk is deemed too great. Risk factors might involve development cost overruns, operating-cost miscalculation, or any other factor that could, in the customer's judgment, result in a less-than-satisfactory final product.
- The existing prototype is evaluated in the same manner as was the previous prototype, and if necessary, another prototype is developed from it according to the fourfold procedure outlined above.

- The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.
- The final system is constructed, based on the refined prototype.
- The final system is thoroughly evaluated and tested. Routine maintenance is carried on a continuing basis to prevent large scale failures and to minimize down time

The following diagram shows how a spiral model acts like:

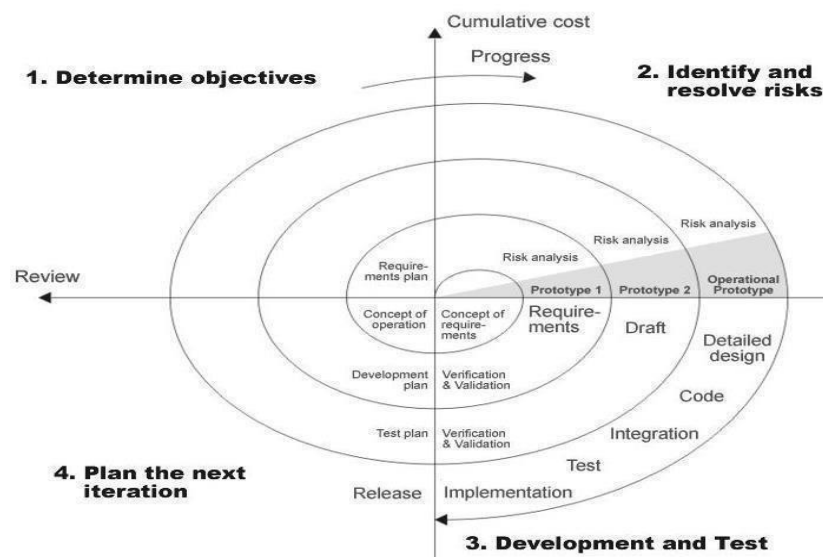


Figure: 3.1 Spiral Model

Advantages

- Estimates(i.e. budget, schedule etc .) become more realistic as work progresses, because important issues discovered earlier.
- It is more able to cope with the changes that are software development generally entails.
- Software engineers can get their hands in and start working on the core of a project earlier.

3.6 Project Implementation Plan

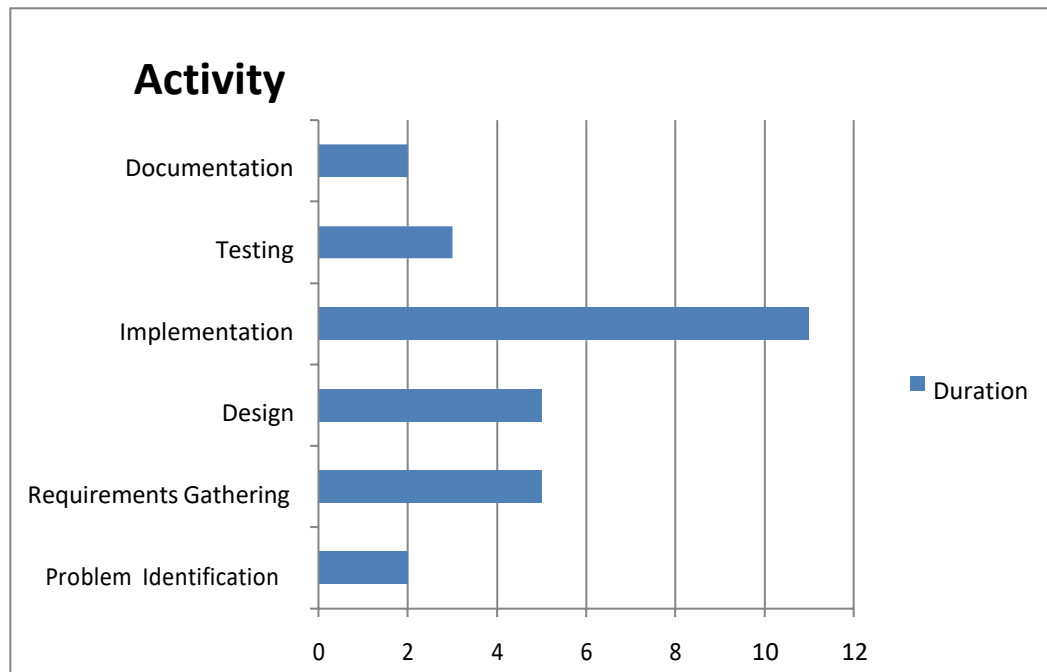


Figure: 3.2 Project Implementation Plan

SOFTWARE REQUIREMENT SPECIFICATION

4. SOFTWARE REQUIREMENT SPECIFICATIONS

4.1 Functional Requirements

- The farmer is able to register his data in the blockchain.
- The farmer should able to enter product details.
- The customer is able to check the product and farmer details.
- If the customer wants to fund the farmer he can able to do it.
- Wallet related keys are need to kept securely.

4.2 Non-Functional Requirements

The major non-functional Requirements of the system are as follows

1. **Usability:** The system is designed in a way where the user needs to put the image information.
2. **Reliability:** The system is more reliable because of the qualities that are inherited from the chosen python platform. The code built by using python is more reliable.
3. **Accessibility:** It can be easily accessible, that is click and run.
4. **Efficiency:** Resources consumption for given load is quite low.
5. **Fault tolerance:** Our system is fault tolerant due to insufficient hardware and Training.
6. **Robustness:** Our system is not capable to cope with errors during execution.
7. **Scalability:** Our project is scalable, that is we can add more resources to our project without disturbing the current scenario

4.3 Software Requirement Specifications

- Ganache : v7. 5. 0
- Truffle : v5. 6. 6
- Nodejs : v18. 12. 1
- Web3.js : v1. 7. 4
- Operating System : Windows11
- Tool : VS Code and Remix Online IDE
- Language : Solidity (v0. 5. 16)

4.4 Hardware Requirement Specifications

- System Type: 64-bit Operating System, I3 Processor.
- Hard Disk: 5 GB.
- RAM : 2 GB and more.
- 3 Mega pixel Camera
- Monitor : 15 inch VGA Colour
- Mouse : Hp Mouse
- Keyboard : Standard Keyboard

SYSTEM DESIGN

5. SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

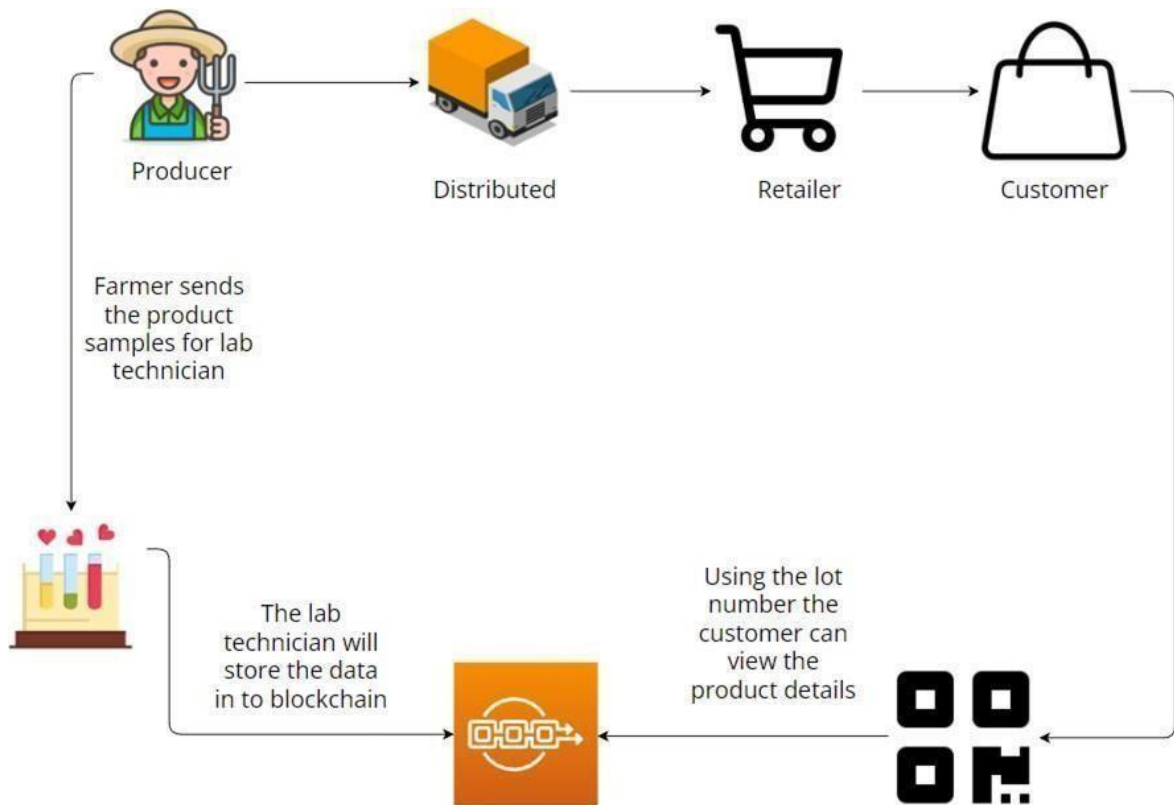


Figure: 5.1 System Architecture

Once the user uploads the image to the system, the system will process the image and send the image to the Character recognition phase where the k-NN algorithm has been already trained with different styles. Now, with all the learnings and experiences the k-NN model will try to classify the text on the number plate. Later the text can be saved in database.

5.2 DESIGN TOOL USED – VISUAL PARADIGM TOOL

Introduction

Visual Paradigm (VP-UML) is a UML CASE Tool supporting UML2, SysML and

Business Process Modeling Notation (BPMN) from the Object Management Group (OMG). In addition, modeling support, it provides report generation and code engineering capabilities including code generation. It can reverse engineer diagrams from code, and provide round-trip engineering for various programming languages.

Contents

- Product Editions
- UML Modeling
- Requirements Management
- Business Process Modeling
- Data Modeling

Product Editions

Higher-priced editions provide more features.

The following editions were available:

- Community Edition
- Modeler Edition
- Standard Edition
- Professional Edition
- Enterprise Edition

UML MODELING

Visual Paradigm supports 13 types of diagrams:

- Class Diagram
- Use Case Diagram
- Sequence Diagram
- Communication Diagram
- State Machine Diagram
- Activity Diagram
- Component Diagram

Requirements Management

- Visual Paradigm supports requirements management including user stories, usecases, SysML requirement diagrams and textual analysis.
- A SysML requirement diagram specifies the capability or condition that must be delivered in the target system. Capability refers to the functions that the system must support. Condition means that the system should be able to run or produce the result given a specific constraint. Visual Paradigm provides a SysML requirement diagram for specifying and analyzing requirements

Business Modeling

Supports BPMN 2.0 for modeling of business processes. The latest version (Aug2016) also supports Case Management with CMMN

Data Modeling

Visual Paradigm supports both Entity Relationship Diagrams (ERD) and Object Relational Mapping Diagrams (ORMD). ERD is used to model the relational database. ORMD is one of the tools to show the mapping between class from object-oriented world and entity in relational database world.

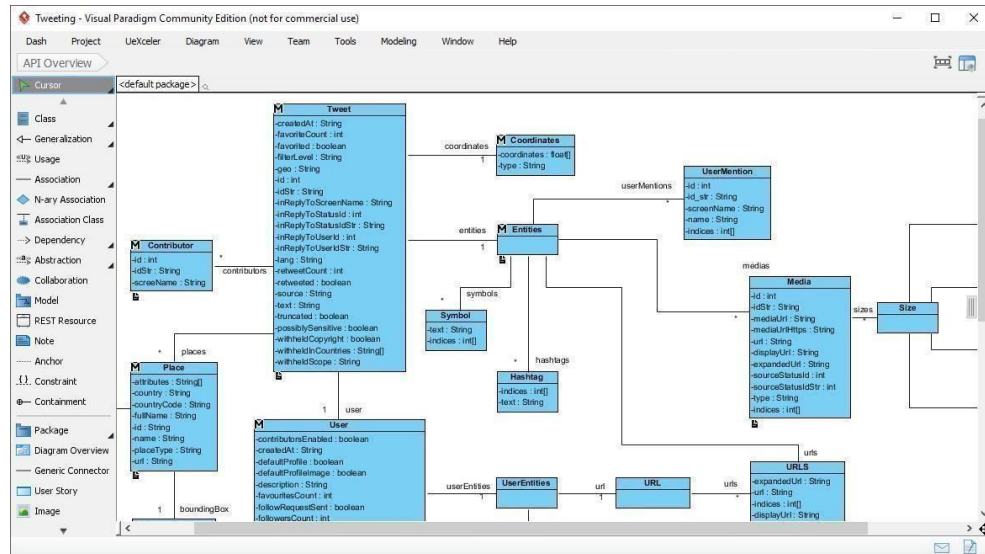


Figure: 5.2 Interface of Visual Paradigm

5.3 UML DIAGRAMS

5.3.1 USECASE DIAGRAM :

It shows the set of use cases, actors & their relationships. In our project we have 1 actor and 1 system where the user needs to input the image to the system and the system processes the image and outputs the text on the number plate.

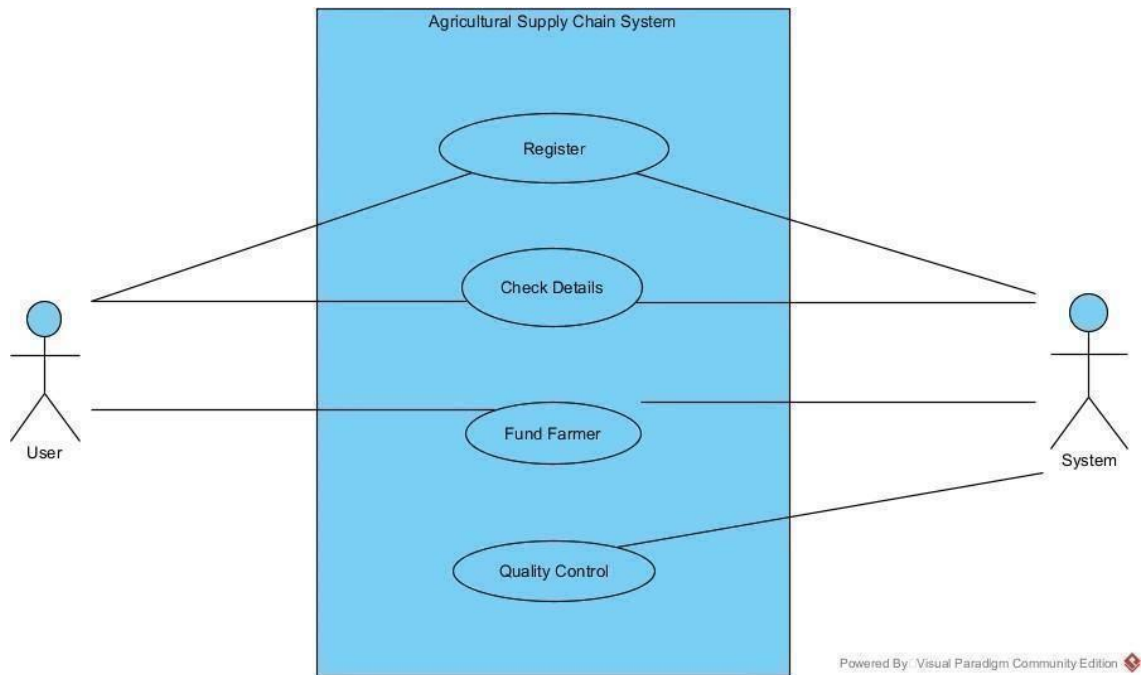


Figure: 5.3 Use Case Diagram

5.3.2 CLASS DIAGRAM:

A *class diagram* shows a set of classes, interfaces, and collaborations and their relationships. These diagrams are the most common diagram found in modeling object-oriented systems. Class diagrams address the static design view of a system. Class diagrams that include active classes address the static process view of a system.

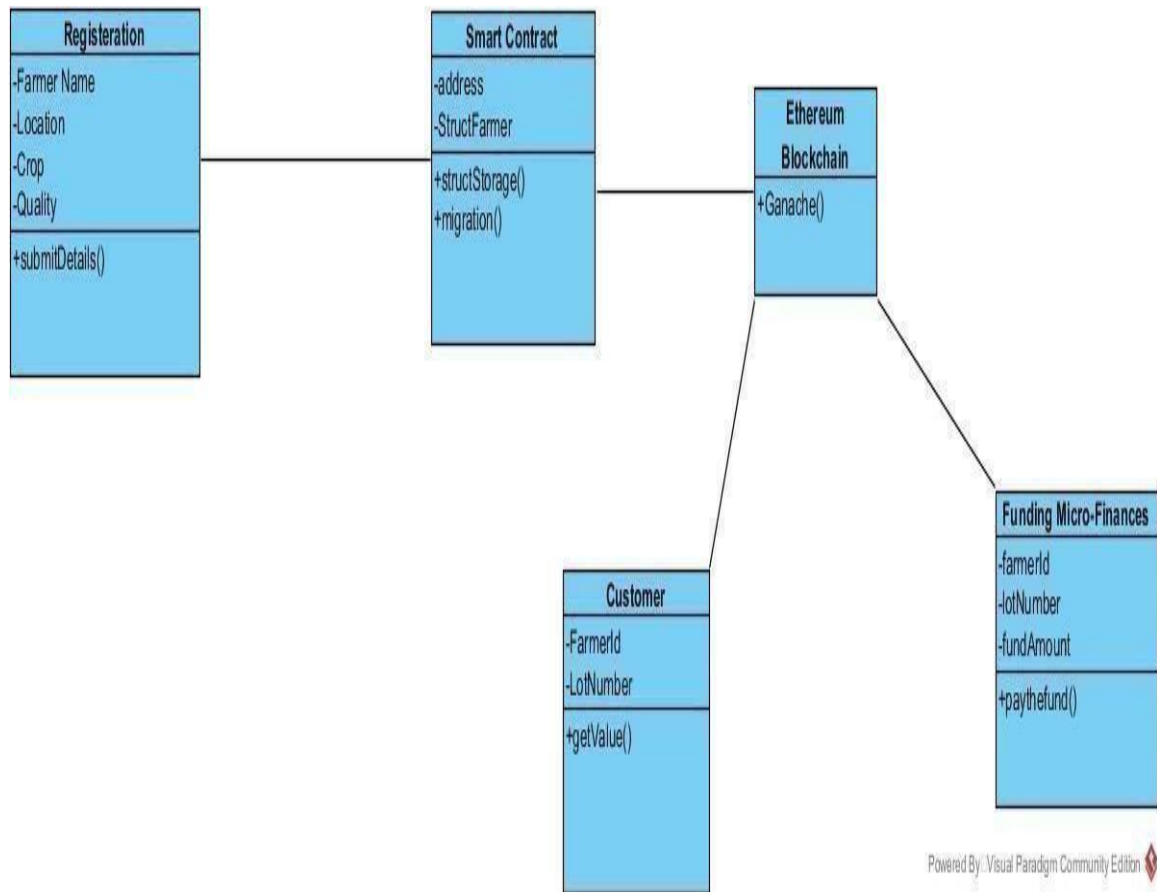


Figure: 5.4 Class Diagram

5.3.3 SEQUENCE DIAGRAM:

A *sequence diagram* is an interaction diagram that emphasizes the time-ordering of messages; a *collaboration diagram* is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Sequence diagrams and collaboration diagrams are isomorphic, meaning that you can take one and transform it into the other.

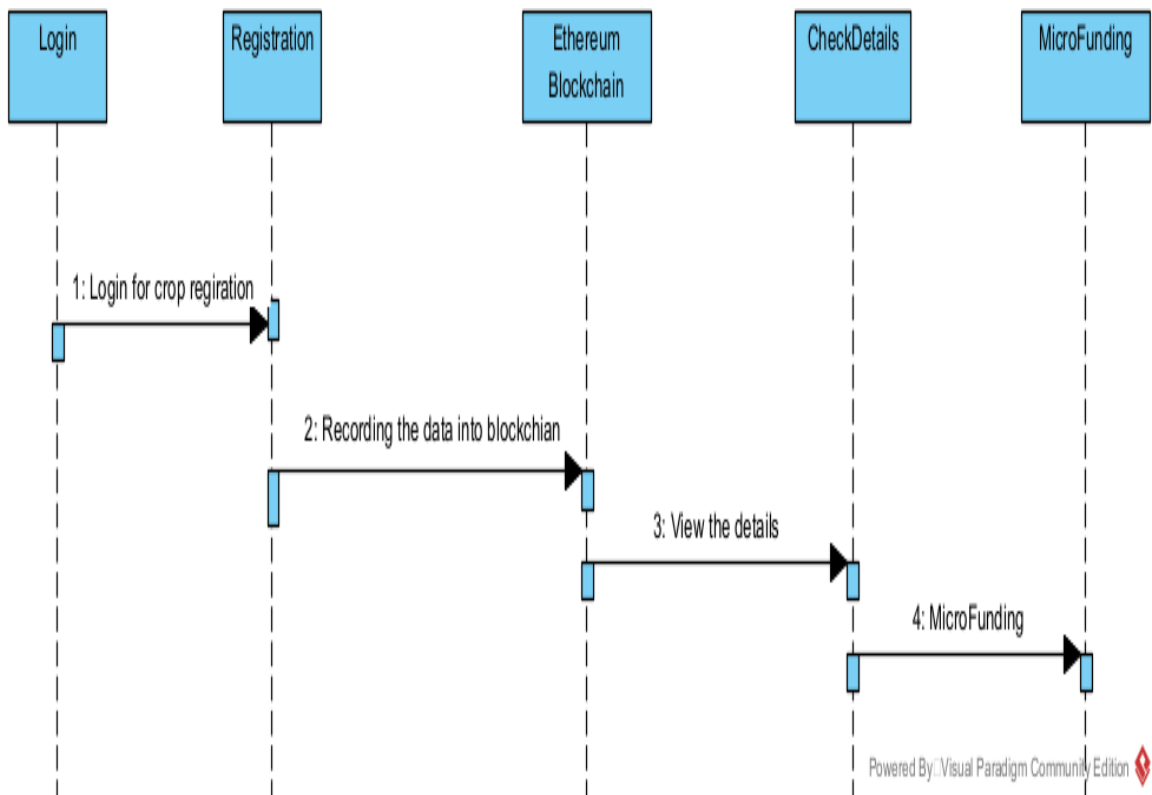


Figure: 5.5 Sequence Diagram

5.3.4 ACTIVITY DIAGRAM:

An *activity diagram* is a special kind of a state chart diagram that shows the flow from activity to activity within a system. Activity diagrams address the dynamic view of a system. They are especially important in modeling the function of a system and emphasize the flow of control among objects.

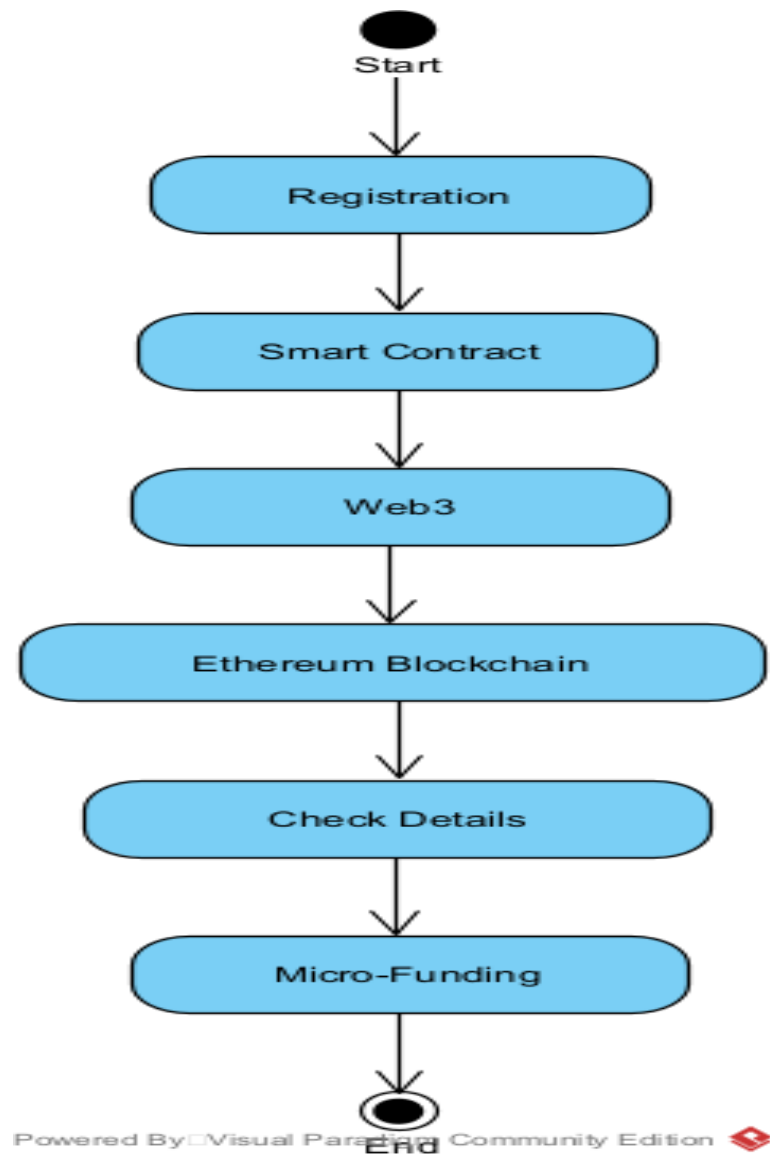


Figure: 5.6 Activity Diagram

IMPLEMENTATION

6. IMPLEMENTATION

6.1 INTRODUCTION

Software Environment:

The software environment involved in blockchain technology typically includes a decentralized network of nodes that work together to maintain a shared ledger of transactions. This network is secured using cryptography and consensus algorithms to ensure the integrity of the ledger. The most common programming languages used for developing blockchain applications are Solidity, JavaScript, and Go. Smart contracts are a key feature of blockchain technology, which are self-executing contracts that automatically enforce the terms of an agreement between parties. Additionally, various blockchain platforms and frameworks such as Ethereum, Hyperledger Fabric, and Corda provide developers with tools and resources to create decentralized applications (dApps) on top of blockchain technology.

Solidity Programming Language

- Solidity is a programming language specifically designed for creating smart contracts on the Ethereum blockchain.
- Solidity is a high-level language, which means that it's easier to read and write than low-level languages like Assembly.
- Solidity is statically typed, which means that the data types of variables must be defined before they can be used.
- Solidity supports object-oriented programming concepts such as inheritance, encapsulation, and polymorphism.
- Solidity has a built-in unit testing framework called "Truffle" that allows developers to test their smart contracts before deploying them.
- Solidity is compiled into Ethereum Virtual Machine (EVM) bytecode, which is then executed on the Ethereum network.

- Solidity has a growing ecosystem of tools and libraries that make it easier to develop and deploy smart contracts.
- Solidity is a Turing-complete language, which means that it can be used to write any kind of program that can be computed.
- Solidity is an open-source language, which means that anyone can contribute to its development and improvement.
- Solidity has its own development environment called "Remix," which provides a web-based interface for writing, testing, and deploying smart contracts.

History of Solidity

Solidity was created by Gavin Wood, Christian Retwisted, and others as part of the development of the Ethereum blockchain. The first version of Solidity was released in August 2014, along with the initial version of the Ethereum platform.

The goal of Solidity was to provide a high-level programming language that could be used to write smart contracts on the Ethereum blockchain. It was designed to be easy to read and write, while still being powerful enough to express complex business logic and financial transactions.

Over the years, Solidity has evolved to include new features and improvements, such as support for contract inheritance, library contracts, and function modifiers. Solidity has also been audited and tested extensively to ensure its security and reliability.

Today, Solidity is widely used by developers and organizations building decentralized applications on the Ethereum network. It has become one of the most popular smart contract languages in the blockchain industry, and continues to be improved and refined by its development community.

Smart Contracts:

Smart contracts are self-executing contracts with the terms of the agreement between buyer and seller being directly written into lines of code. They allow for transparent and

automated execution of transactions without the need for intermediaries, reducing costs and increasing efficiency. Smart contracts are a fundamental technology in blockchain and have the potential to revolutionize the way we conduct business.

NodeJS:

Node.js is an open-source, server-side runtime environment that allows developers to run JavaScript on the server. It is built on the V8 JavaScript engine, which is the same engine used by Google Chrome, and provides a fast and efficient way to run JavaScript code. Node.js is widely used for building web applications, APIs, and microservices.

Ethereum Virtual Machine (EVM):

The Ethereum Virtual Machine (EVM) is a virtual machine that runs on the Ethereum blockchain. It is the runtime environment for executing smart contracts written in high-level programming languages such as Solidity, Vyper, and others.

The EVM is a completely isolated environment, meaning that it does not have access to the underlying hardware of the computer it is running on. This is done for security reasons, as it ensures that smart contracts can run in a trustless environment without being able to affect the host computer.

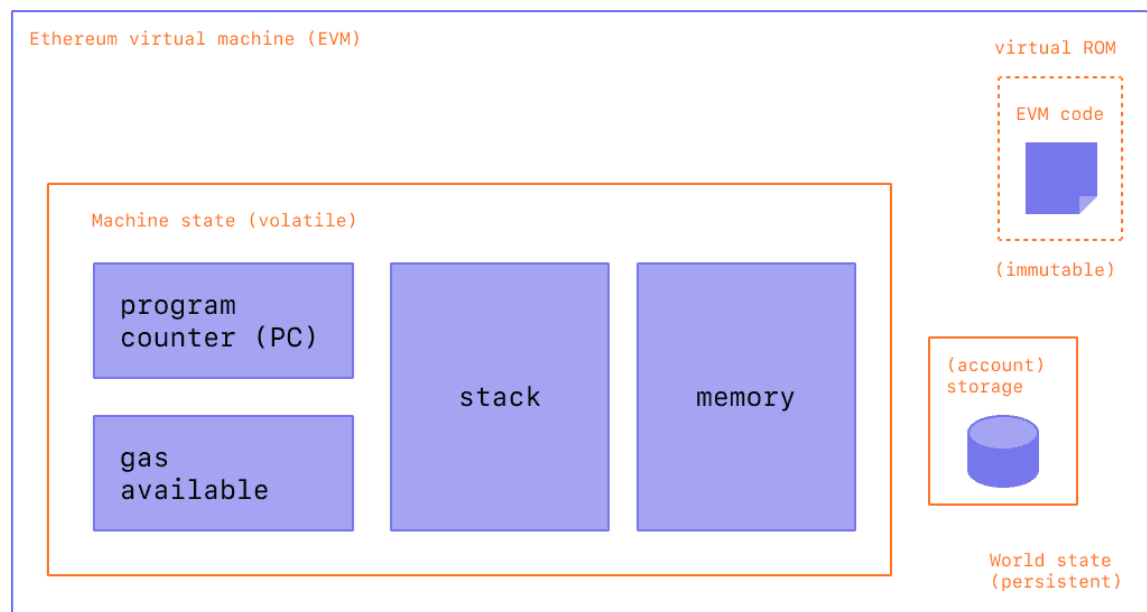


Figure: 6.1 Ethereum Virtual Machine

The EVM uses its own bytecode format, which is generated by compiling high-level programming languages such as Solidity. This bytecode is then deployed to the Ethereum network, where it can be executed by any node on the network.

The EVM is responsible for executing smart contract code, handling gas fees, and maintaining the state of the Ethereum blockchain. It is designed to be highly fault-tolerant, meaning that it can continue running even in the face of errors or malicious behavior.

One of the key features of the EVM is that it is Turing-complete, which means that it can execute any program that can be computed. This allows developers to write complex smart contracts that can perform a wide range of tasks, from simple transfers of cryptocurrency to more complex financial transactions and logic.

Overall, the Ethereum Virtual Machine is a crucial component of the Ethereum blockchain, as it provides a secure and reliable environment for executing smart contracts and maintaining the integrity of the network.

Solidity Language Features

Solidity Language's features include

- **Smart Contract Development:** Solidity is specifically designed for developing smart contracts on the Ethereum blockchain. It provides a high-level syntax that allows developers to express complex business logic and financial transactions in a simple and readable manner.
- **Object-Oriented Programming:** Solidity supports object-oriented programming concepts such as inheritance, encapsulation, and polymorphism, which makes it easier to write modular and maintainable code.
- **Statically Typed:** Solidity is a statically typed language, meaning that variables and their data types must be explicitly declared before they can be used. This helps catch errors and make the code more predictable.
- **Turing-Complete:** Solidity is a Turing-complete language, which means that it can be used to write any kind of program that can be computed.

- **Gas Fees:** In order to execute code on the Ethereum blockchain, gas fees must be paid. Solidity includes a gas cost estimator, which helps developers optimize their code to minimize gas usage and reduce costs.
- **Debugging:** Solidity provides built-in debugging tools that help developers identify and fix errors in their code.
- **Contract Verification:** Solidity contracts can be verified on the Ethereum blockchain using the Etherscan service, which allows users to confirm that the source code of a contract matches its compiled bytecode.
- **Integration with Ethereum Tools:** Solidity is designed to integrate with other Ethereum tools and libraries, such as the Truffle development framework and the Remix IDE.

6.2 TECHNOLOGIES USED

We have used few technologies such as Ganache, Truffle, MetaMask . They have explained in detail below:

6.2.1 MetaMask

MetaMask is a non-custodial wallet, meaning that users have complete control over their private keys and funds. This provides a high level of security and eliminates the need for users to trust a centralized third-party to hold and manage their cryptocurrency. Additionally, MetaMask is an open-source project, which means that anyone can audit and contribute to its codebase.

One of the key features of MetaMask is its ease of use. The wallet is designed with a user-friendly interface that makes it easy for newcomers to get started with Ethereum and decentralized applications. MetaMask allows users to quickly create new Ethereum accounts, switch between accounts, and manage their account settings. Users can also import existing accounts from other wallets using their private keys or seed phrases.

Another important feature of MetaMask is its ability to interact with decentralized applications (dApps) on the web. MetaMask injects a Web3 provider into

the browser, which allows dApps to communicate with the Ethereum network and access the user's wallet. This enables users to seamlessly interact with dApps, such as decentralized exchanges, lending protocols, and more, without needing to manually enter their wallet information for each transaction.

In addition to its browser extension and mobile app, MetaMask also provides a developer API that allows developers to integrate MetaMask with their own dApps. This makes it easier for developers to build and deploy decentralized applications that can interact with the Ethereum network and leverage the security and ease-of-use features of MetaMask.

Overall, MetaMask has become a popular choice for cryptocurrency users and developers alike, due to its security, ease-of-use, and interoperability with the Ethereum ecosystem.

6.2.2 Ganache

Ganache is a popular personal blockchain for Ethereum development that allows developers to create and test smart contracts on a local blockchain environment. It is often used during the development phase of Ethereum projects as it allows developers to quickly and easily test their code without incurring gas fees or waiting for transactions to be mined on the live Ethereum network.

Ganache provides a local blockchain that can be run on a developer's computer, allowing for rapid testing and debugging of smart contracts. It comes with a built-in Ethereum client, which allows developers to deploy and interact with smart contracts using web3.js, a popular JavaScript library for Ethereum development.

In addition to its core functionality, Ganache also provides a number of useful features for Ethereum development, such as the ability to simulate different network conditions, control mining behavior, and view transaction logs and debugging information. Ganache also includes a user-friendly interface that makes it easy for developers to manage their local blockchain and interact with their contracts.

Ganache is available in both a free, open-source version and a paid, enterprise version with additional features and support. It is widely used by Ethereum developers and is considered an essential tool for developing and testing smart contracts.

6.2.3 Truffle

Truffle is a popular development framework for building decentralized applications (dApps) on the Ethereum blockchain. It provides a suite of tools that streamline the development process and make it easier for developers to create, test, and deploy smart contracts.

Truffle includes a number of core components, such as the Truffle Contract library, which allows developers to write smart contracts in Solidity, the most popular programming language for Ethereum, and interact with them using JavaScript. Truffle also includes a development blockchain called Ganache, which can be used for local testing and debugging.

One of the key features of Truffle is its ability to automate many of the tasks involved in smart contract development, such as compiling code, managing dependencies, and deploying contracts. Truffle also provides a built-in testing framework that makes it easy for developers to write and run tests for their contracts.

Truffle's user interface, Truffle Suite UI, provides a user-friendly interface for managing projects, compiling and deploying contracts, and interacting with the Ethereum network. Truffle also integrates with a number of popular tools and services in the Ethereum ecosystem, such as MetaMask, Infura, and IPFS.

Truffle is open-source and has a large and active community of developers contributing to its development. It is widely used by Ethereum developers and is considered one of the most powerful and comprehensive tools for building dApps on the Ethereum blockchain.

6.3 Coding Standards

DMAIC is a data driven quality strategy used to improve processes. It is an integral part of six sigma initiative, but in general can be implemented as a standalone quality improvement procedure or as part of other process improvement initiatives such as lean.

DMAIC is an acronym for the five phases that make up the process:

- **Define** the problem, improvement activity for improvement, the project goals, and customer requirements.
- **Measure** process performance
- **Analyze** the process to determine root causes of variation ,poor performance
- **Improve** process performance by addressing and eliminating the root causes
- **Control** the improve process and future process performance

Define, measure, analyze, improve and control (DMAIC) is a structured problem-solving method. Each phase builds on the previous one, with the goal of implementing long-term solutions to problems. Sometimes, project leaders or sponsors don't feel a formal approach is necessary, but most problem-solving efforts benefit from a disciplined method.

The tools used in the define phase lay the foundation for the project. The team accurately and succinctly defines the problem, identifies customers and their requirements, and determines skills and areas that need representation on the project team. Individuals who must be part of the core team or be ad-hoc members are identified, and project measures, financials and a communication plan are established.

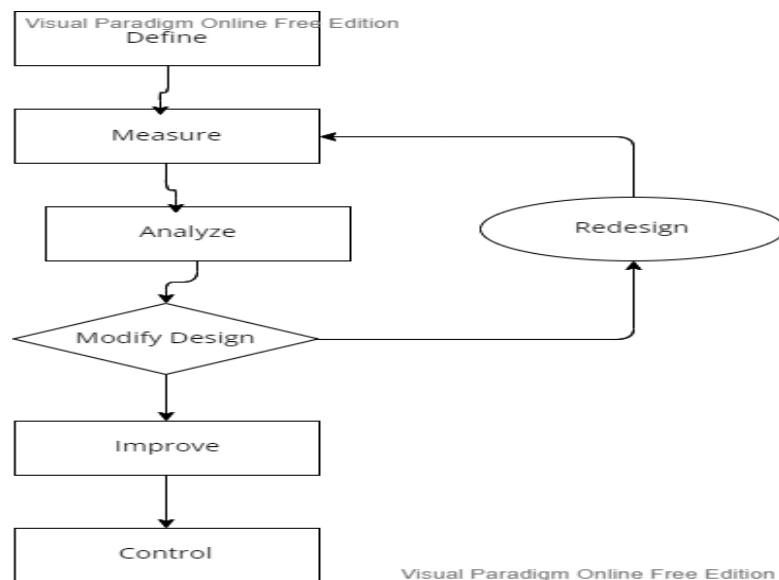


Figure: 6.2 DMAIC

The measure phase is when the true process is identified and documented. Process steps, and corresponding inputs and outputs are identified. Measurement systems are identified or developed, and validated and improved as required. Baseline performance is established with trustworthy data.

In the analyze phase, the critical inputs are identified. Inputs that have a strong relationship with the outputs and root causes are determined. These critical inputs are the drivers of performance.

In the improve phase, potential solutions are identified and evaluated, and the process is optimized. The critical inputs that must be controlled to maintain performance that reliably satisfies the customer are determined. Process capability and project financials are estimated.

The control phase establishes mistake-proof, long-term measurement and reaction plans. The team develops standard operating procedures and establishes process capability. Project financials are updated, verified and reported. Control plans are established with reaction plans, ownership and control is transitioned to the process owner, and lessons are documented. The team documents opportunities to spread the outcomes to other areas in the organization.

When to use DMAIC

When improving a current process, if the problem is complex or the risks are high, DMAIC should be the go-to method. Its discipline discourages a team from skipping crucial steps and increases the chances of a successful project, making DMAIC a process most projects should follow.

There are two approaches to implementing DMAIC. The first is the team approach in which individuals who are skilled in the tools and method, such as quality or process improvement experts, lead a team. The team members work on the project part-time while caring for their everyday responsibilities. The quality or process improvement expert might be assigned to several projects. These are long-duration projects taking months to complete.

6.4 MODULES

1. Farmer and Product Registration
2. Quality Testing
3. Customer Check Details
4. Micro Finance

6.4.1 FARMER AND PRODUCT REGISTRATION :

The supply chain application includes a registration form for farmers that allows them to enter their personal and farm details. Once the details are entered, they are stored directly onto the blockchain. The technology underlying the application utilizes Truffle, a popular development framework for building decentralized applications on the Ethereum blockchain, for the deployment of smart contracts.

The backend blockchain is powered by go-ethereum (geth), which is one of the most widely used Ethereum clients. In order to interact with the blockchain, the Web3 Javascript provider API is used. This allows the application to communicate with the blockchain and perform various operations such as reading and writing data to the blockchain.

By leveraging Truffle and the Web3 API, the supply chain application is able to securely and efficiently store and manage the data of farmers in the supply chain.

6.4.2 QUALITY CHECKING:

The quality testing section of the application allows users to retrieve farmer details by entering a unique Farmer ID. These details are stored on the blockchain using Solidity code as a data structure. In order to retrieve the farmer details, a mapping data structure is used, which allows for efficient retrieval of data based on unique identifiers.

Within the Quality Testing page, users can view the details of the block where the farmer's information is stored on the blockchain. The 'Approve Details' button can be clicked to approve the details of the farmer, which will then redirect the user to the Product Details page. The Product Details page is also part of the quality testing process, where users can enter information such as lot number, grade, price, test date, and expiry date.

These details are stored on the blockchain as a data structure, using Solidity code, allowing for secure and transparent record-keeping of product information. By utilizing the mapping data structure and Solidity code, the quality testing process is streamlined and provides an efficient way to store and retrieve important data related to the supply chain.

6.4.3 CUSTOMER CHECK DETAILS:

The customer page in the supply chain application allows customers to view their own details and check the status of quality testing for their agricultural produce. To do this, the customer must enter both the farmer ID and lot number to retrieve the relevant information.

This information is stored on the blockchain and can be retrieved using the appropriate code. By allowing customers to access this information, the supply chain application promotes transparency and provides a convenient way for customer to say

informed about the quality and status of their produce. With this feature, customers can have greater trust in the supply chain and feel more connected to the production process.

6.4.4 MICRO FINANCE:

The micro-finance form is a feature within the supply chain application that allows any user to provide funding to a farmer. To do this, the user must provide the farmer's public ID, the lot number of the product, and the desired amount of funding.

This feature is designed to provide a way for users to support farmers in need, while also promoting the growth and development of the agricultural sector. By enabling micro-funding, the supply chain application can help farmers access the resources they need to produce high-quality products and promote sustainable practices. With this feature, users can feel empowered to make a positive impact in the community, while also supporting the growth of the agricultural industry.

SYSTEM TESTING

7. SYSTEM TESTING

7.1 TESTING PLAN

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

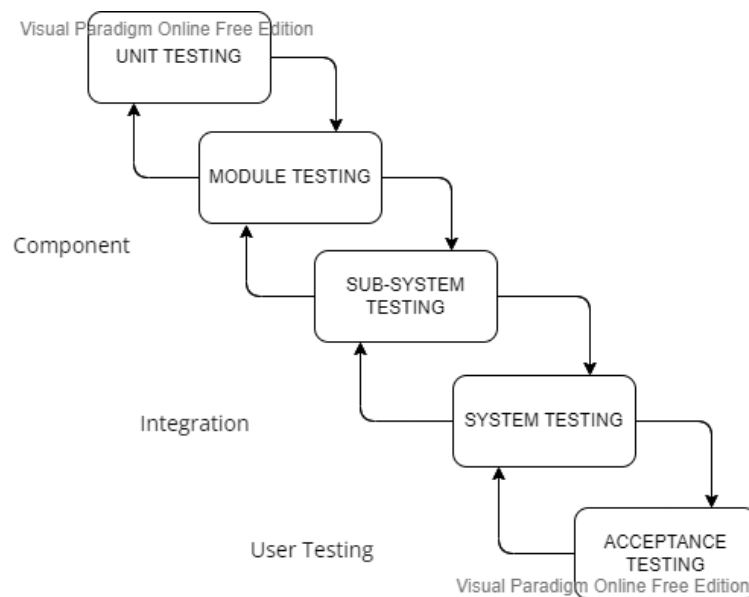


Figure: 7.1 Testing Plan

7.2 SOFTWARE TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.2.1 TESTING METHODOLOGIES

The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**
- **Black box Testing**
- **White box Testing**

7.2.2 Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing. During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

7.2.3 Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

7.2.4 User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

7.2.5 Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required

output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

7.2.6 Validation Testing

Validation checks are performed on the following fields.

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested. A successful test is one that gives out the defects for the inappropriate data and produces an output revealing the errors in the system.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

7.2.7 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

7.2.8 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

TESTING STRATEGY:

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements. Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing

7.3 TEST CASES:

| Test Case ID | Testcase Description | Expected Results | Actual Result | Result |
|--------------|---|--|---|--------|
| 01 | Farmer and Product Registration interface | The farmer should be able to successfully register the product details | The Farmer is being able to successfully register the product details | Pass |
| 02 | Quality Check | The lab technical should be able to check the product quality and able to record the details and also should be able to approve the product. | The lab technical is being able to check the product quality and able to record the product details and also able to approve the product. | Pass |
| 03 | Customer Check Details | The customer should be able to check the product details with the help of farmer ID and Lot number of the product. | The customer is able to check the product details with the help of farmer ID and Lot number of the product. | Pass |
| 04 | Micro-Finance | The customer should be able pay micro- finances to the farmer. | The customer is being able pay micro-finances to the farmer. | Pass |

Table 1: Unit Test Cases

7.4 BUG REPORT

| S.no | Steps | Description |
|------|-----------------|--------------------------------------|
| 1. | Bug_id | Bug_1 |
| 2. | Bug Description | Connection with the Ethereum account |
| 3. | Expected output | Visibility of Public ID |
| 4. | Actual output | Public ID is visible |
| 5. | Priority | High |
| 6. | Severity | High |

Table 2: Bug Report – 1

| S.no | Steps | Description |
|------|-----------------|---|
| 1. | Bug_id | Bug_2 |
| 2. | Bug Description | Connection between MetaMask and Ganache accounts |
| 3. | Expected output | Visibility of Ethereum balance in the MetaMask wallet |
| 4. | Actual output | Ethereum balance is visible and same reflected in the ganache account |
| 5. | Priority | High |
| 6. | Severity | High |

Table 3: Bug Report – 2

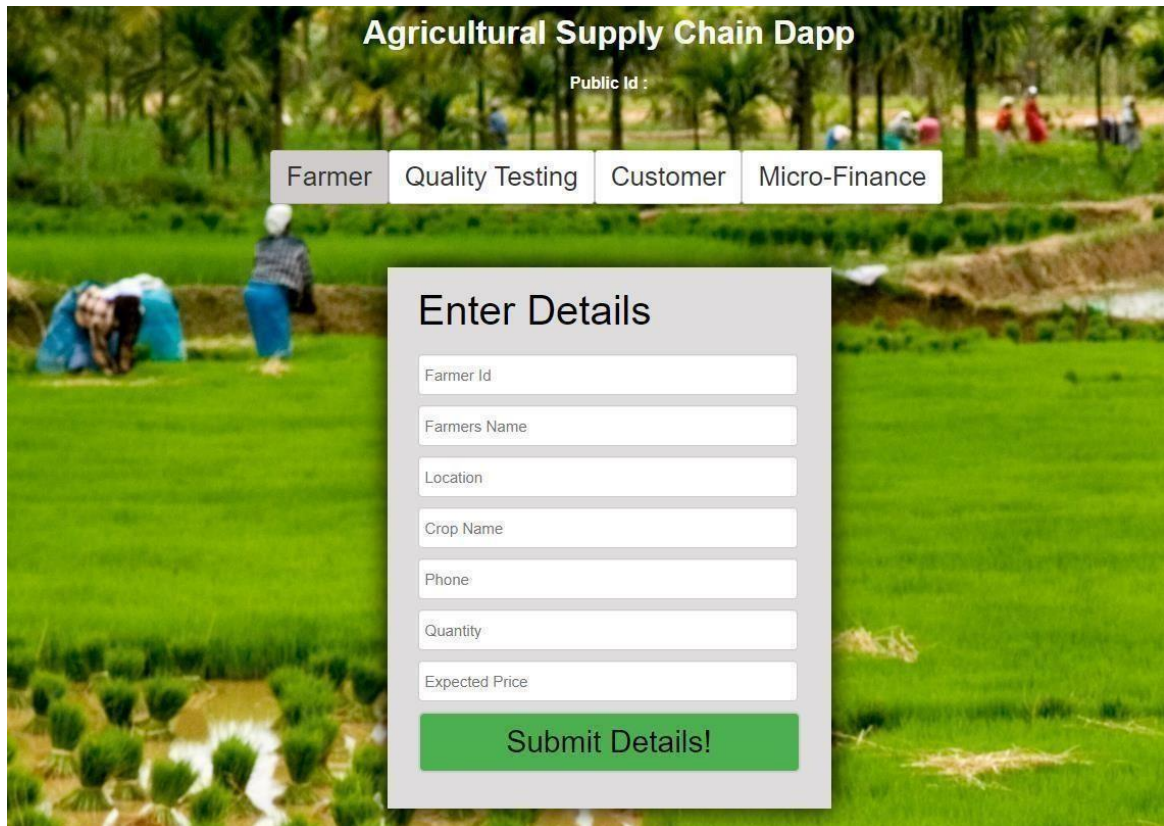
| S.no | Steps | Description |
|------|-----------------|---|
| 1. | Bug_id | Bug_1 |
| 2. | Bug Description | Occurring of the all the transactions |
| 3. | Expected output | Visibility of all the transactions in MetaMask wallet |
| 4. | Actual output | All the transaction are properly visible |
| 5. | Priority | High |
| 6. | Severity | High |

Table 4: Bug Report - 3

RESULT SCREENS

8. RESULT SCREENS

Product registration Interface – 1



Agricultural Supply Chain Dapp

Public Id :

Farmer | Quality Testing | Customer | Micro-Finance

Enter Details

Farmer Id

Farmers Name

Location

Crop Name

Phone

Quantity

Expected Price

Submit Details!

Figure: 8.1 Product registration Interface - 1

Navigation:

- To execute the project first we need to open ganache, and connect the ganache account to MetaMask.
- Now run the command: `truffle migrate`, with this the smart contracts has been deployed.
- Replace the contract address at `src>app.js` file.
- Now run command: `npm run dev`, with this application get starts.
- You will be given with localhost open it to view the other functions.
- The farmer can able to register the product through the given forum.

Quality Testing Interface- 2

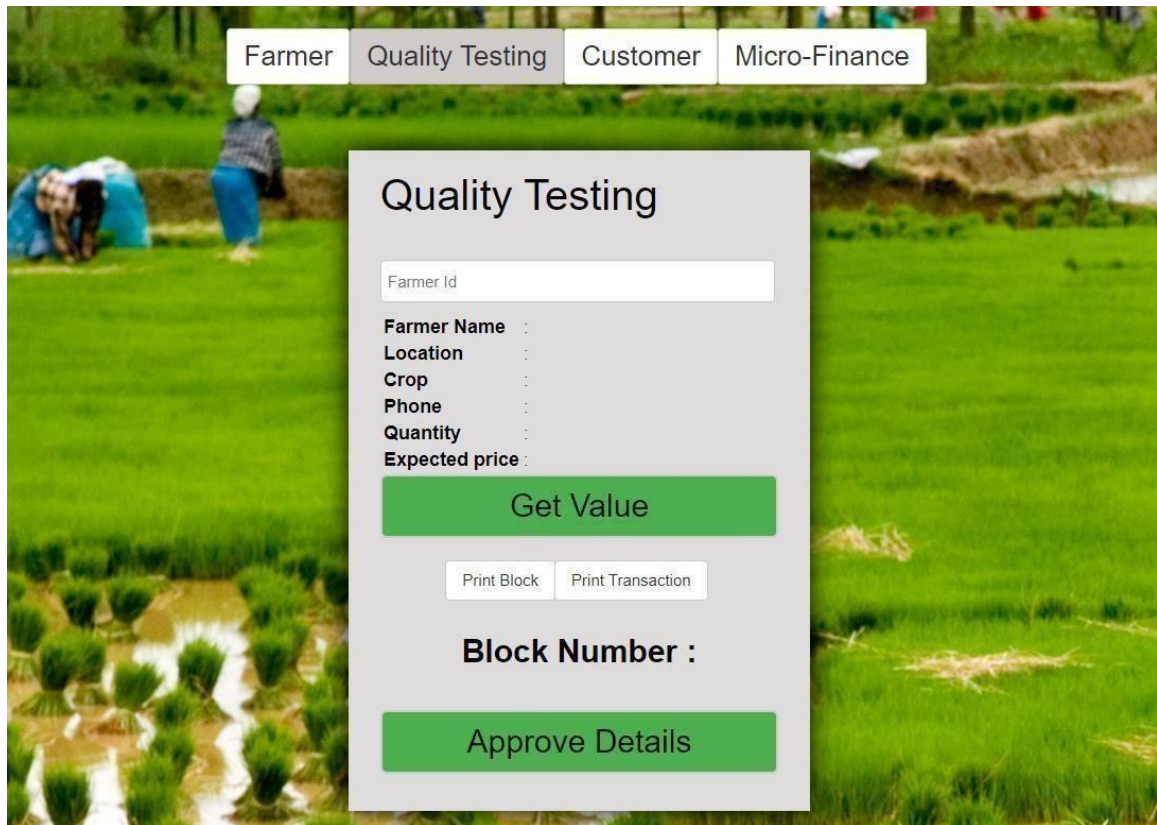
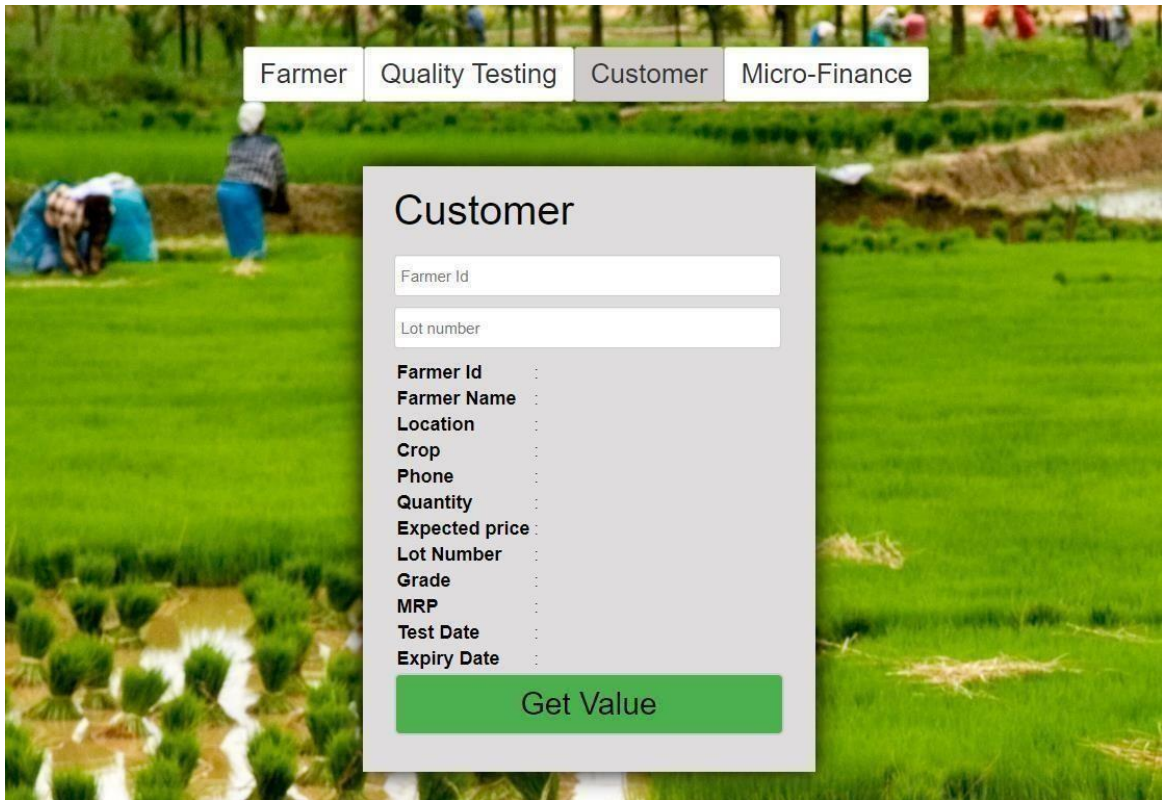
The image shows a web application interface for quality testing, overlaid on a background photograph of a lush green rice paddy field with people working. At the top, there is a navigation bar with four tabs: 'Farmer', 'Quality Testing' (which is highlighted), 'Customer', and 'Micro-Finance'. The main content area is a light gray box titled 'Quality Testing'. It contains a text input field for 'Farmer Id'. Below this are several labels followed by colons: 'Farmer Name', 'Location', 'Crop', 'Phone', 'Quantity', and 'Expected price'. These labels are positioned to the left of a large green button labeled 'Get Value'. Below the 'Get Value' button are two smaller, light gray buttons: 'Print Block' and 'Print Transaction'. Further down is the label 'Block Number :' followed by another large green button labeled 'Approve Details'.

Figure: 8.2 Quality Testing

Navigation:

- Lab Technicians is able to view and approve the product details.
- Here the lab technicians is able to assign the lot number to the product.

Customer Interface – 3



The screenshot displays a web application interface for a customer. At the top, there are four navigation tabs: 'Farmer', 'Quality Testing', 'Customer' (which is highlighted), and 'Micro-Finance'. Below these tabs, a modal window titled 'Customer' is open. This modal contains two input fields: 'Farmer Id' and 'Lot number'. Below the input fields is a list of product details, each followed by a colon, indicating where the information will be displayed: Farmer Id, Farmer Name, Location, Crop, Phone, Quantity, Expected price, Lot Number, Grade, MRP, Test Date, and Expiry Date. At the bottom of the modal is a green button labeled 'Get Value'. The background of the application is a photograph of a lush green rice field with people working in the distance.

Figure: 8.3 Customer Interface

Navigation:

- The customer should be able to view the product details.
- With the help of Farmer Id and Lot Number, the customer should be able to view the product details.

Micro-Finance Interface – 4



Figure: 8.3 Micro- finance Interface

Navigation:

- The customer should be able to pay the micro finances to the farmer.
- All these transaction can be seen on the MetaMask wallet.

CONCLUSION & FUTURE SCOPE

9. CONCLUSION AND FUTURE SCOPE

In the existing system, all the details related to the product are recorded on a physical book with which we had many issues like losing the record, not having proper security for the data and many issues with the traceability and transparency.

But we can reduce these issues with the help of blockchain technology that could give good security to the records and data, ease in traceability with good level of transparency. From the farmer's perspective, he may demonstrate the caliber of his goods, bargain appropriately, and earn healthy profits.

From the standpoint of the retailer and the customer, they can distinguish between healthy and unhealthy products and can confidently purchase the healthy products. By providing additional funding, the customer can also fund or purchase directly from the farmer. The relationship of trust between the farmer, retailer, and customer will improve as a result.

As a future enhancement, there are several areas where we can think of applying blockchain technology such as voting system, managing the data of huge organizations.

BIBILOGRAPHY

10. REFERENCES

- [1] Agricultural Supply Chain Management Using Blockchain Technology. Published Year:2020,Author: Bhagya Hegde et al
- [2] A Theoretical Implementation: Agriculture- Food Supply Chain Management using Blockchain Technology, Published Year: 2019,Author: S. Madumidha et al
- [3] Blockchain-Based Agri-Food Supply Chain: A Complete Solution, Published Year:2020,Author: AFFAF SHAHID et al
- [4] Blockchain technology in current agricultural systems: from techniques to applications,Published Year: 2020,Author: WEIJUN LIN et al.
- [5] Blockchain-Based Soybean Traceability in Agricultural Supply Chain, Published Year:2019,Author: KHALED SALAH et al
- [6] P.Helo and B.Szekely,"Logistics information systems: An analysis of software solutions for supply chain coordination ,"Industrial management &Data Systems,vol.105, no. ,1,pp, 5-18,2005.
- [7] V. Buterin ,"A next generation smart contract and decentralised application platform
,"Ethereum project white paper ,2014.
- [8] K .Wust and A. Gervais ,"do you need a blockchain ?" cryptology ePrint archive, report2017/375,2017.
- [9] Ajay N, Anitha N," Supply Of Agriculture Product By Ensuring Quality through Blockchain Technology".
- [10] A Zebra technologies White Paper," Barcoding and RFID Enable Food Supply chain Traceability and Safety" Available at:
<https://www.abr.com/wpcontent/uploads/2014/04/Zebra-food-traceability-enus.pdf>
- [11] Mischa Tripoli ,Josef schemidhuber ,"Emerging Opportunities For the Application Of Blockchain in the Agri-Food industry".
- [12] Ramachandran and K. Murat, "Using blockchain and smart contracts for secure data provenance management," arXiv preprint arXiv:1709.10000, 2017.

- [13] S. Srinivasan, D. Shanthi, and A. Anand, "Inventory transparency for agricultural produce through IoT," IOP Conf. Ser.: Mater. Sci. Eng, vol. 211, no. 1, p. 012009, 2017.
- [14] Tian, "An agri-food supply chain traceability system for china based on rfid and blockchain technology," in Proc. of the ICSSSM. IEEE, 2016, pp. 1–6.
- [15] Tian, "A supply chain traceability system for food safety based on HACCP, blockchain and Internet of Things," in Proc. of the ICSSSM, 2017, pp. 1–6.

APPENDIX

11. APPENDIXES

11.1 SAMPLE CODE

//StructStorage.sol:

```
pragma solidity ^0.5.0;
contract StructStorage {
    uint256 public s = 1;
    uint256 public c;
    uint256 public t=1;
    mapping (address => uint) balances;
    function fundaddr(address addr) public{
        balances[addr] = 2000;
    }
    function sendCoin(address receiver, uint amount, address sender) public
returns(bool sufficient){
    if (balances[sender] < amount)
        return false;

    balances[sender] -= amount;
    balances[receiver] += amount;

    return true;}
    function getBalance(address addr) view public returns(uint){
        return balances[addr];
    }
}

struct farmer {

    bytes fid;
    bytes32 fname;
    bytes32 loc;
    bytes32 crop;
    uint256 contact;
    uint quantity;
```

```

        uint exprice;
    }
    struct lot {

        bytes lotno;
        bytes grade;
        uint mrp;
        bytes32 testdate;
        bytes32 expdate;
    }

    address public tester;
    address owner;

    mapping (bytes => farmer) f1;

    return }

farmer[] public fm;
mapping (bytes => lot) l1;
lot[] public l;

function produce(bytes memory id, bytes32 name, bytes32 loc, bytes32 cr, uint256 con,
uint q, uint pr) public{

    StructStorage.farmer memory fnew = farmer(id,name,loc,cr,con,q,pr);
    f1[id] = fnew;
    fm.push(fnew);
    s++;

}

function    getproduce(bytes    memory    j)    public    view    returns(bytes
memory,bytes32,bytes32,bytes32,uint256,uint,uint) {

    return

(f1[j].fid,f1[j].fname,f1[j].loc,f1[j].crop,f1[j].contact,f1[j].quantity,f1[j].exprice);

```

```

    }

    function quality(bytes memory ll, bytes memory g, uint256 p, bytes32 tt, bytes32 e)
    public{

        StructStorage.lot memory lnew=lot(ll,g,p,tt,e);
        ll[ll]=lnew;
        l.push(lnew);
        t++;

    }

```

```

    function getquality(bytes memory k) public view returns(bytes memory,bytes
memory,uint,bytes32,bytes32) {
        return(ll[k].lotno,ll[k].grade,ll[k].mrp,ll[k].testdate,ll[k].expdate);
    }

}

```

// Migrations.sol

```

pragma solidity ^0.5.0;

contract Migrations {
    address public owner;

    uint public last_completed_migration;

    modifier restricted() {

        if (msg.sender == owner) _;

    }

    constructor() public{
        owner = msg.sender;
    }

    function setCompleted (uint completed) restricted public{
        last_completed_migration = completed;
    }
}

```

```
function upgrade(address new_address) restricted public{  
    Migrations upgraded = Migrations(new_address);  
    upgraded.setCompleted(last_completed_migration);  
}}
```