**PATAN MULTIPLE CAMPUS (PMC)**
**Tribhuvan University**
**Institute of Science and Technology**

# ETHEREUM BASED CROWDFARMING PLATFORM WITH SUPPLY CHAIN USING SMART CONTRACT

## A FINAL PROJECT REPORT

**Submitted to**

**Department of Computer Science and Information Technology**

**Patan Multiple Campus**

***In partial fulfillment of the requirements for the Bachelor's Degree in Computer Science and Information Technology***

**Submitted by**

Sagar Subedi
15051/074

Smriti Banjade
15063/074

Shikha Bhatt
16320/074

# SUPERVISOR'S RECOMMENDATION

I hereby recommend that this project be prepared under my supervision by MR. SUNIL LUITEL entitled **"ETHEREUM BASED CROWDFARMING PLATFORM USING SMART CONTRACT"** in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology be processed for the evaluation.

…….

Mr. Sunil Luitel

Lecturer

Patan Multiple Campus

# STUDENT'S DECLARATION

We hereby declare that we are the only author of this work and that no other sources other than that listed here have been used in this work.

Date: 2022/03/20

| | | |
|---|---|---|
| ……..…… | …… | …… |
| Sagar Subedi | Smriti Banjade | Shikha Bhatt |

# LETTER OF APPROVAL

This is to certify that this project prepared by SAGAR SUBEDI, SMRITI BANJADE and SHIKHA BHATT entitled **"ETHEREUM BASED CROWDFARMING PLATFORM USING SMART CONTRACTS" in** partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Information Technology has been well studied. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

| | |
|---|---|
| ..... | ..... |
| Mr. Sunil Luitel<br>Lecturer<br>Patan Multiple Campus | Mr. Mahesh Kumar Yadav<br>Head of Department<br>Patan Multiple Campus |

# ACKNOWLEDGEMENT

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of the project. All that we have done is only due to such supervision and assistance and we would not forget to thank them.

We respect and thank Sunil Luitel for providing an opportunity to do the project and giving all support and guidance which made complete the project duly. We are extremely thankful to him for providing such support and guidance, although he had a busy schedule. We owe deep gratitude to him for his keen interest in our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system.

We are thankful and fortunate enough to get constant encouragement, support and guidance from all the friends and family members who helped us in successfully completing the project work.

| | | |
|---|---|---|
| Name: Sagar Subedi | Name: Smriti Banjade | Name: Shikha Bhatt |
| Roll No.: 15051/074 | Roll No.: 15063/074 | Roll No.: 16320/074 |

# ABSTRACT

Farming takes a lot of effort and time from farmers to produce good quality fruits and vegetables but there's always a fear that haunts farmers of not being able to sell their produce or getting proper price or market. This problem can be solved using the idea of crowdfarming which is a similar concept to crowdfunding. Farmer runs a farming campaign and users can participate by contributing a certain amount of money. This way, farmers can get assured of selling their produce well before the harvest. The crowd farming platform aims to provide such a platform but with the added security and transparency of Blockchain. Firstly, this paper analyzes the existing problem in the process. Then the paper introduces a system that attempts to solve the shortcomings of the current process. In a nutshell, this paper assess the potential of a decentralized architecture using blockchain technology through crowdfarming platform with agricultural supply chain which maintains trust and transparency and also provides authentic information to the end users.

The solution presented is a platform that allows general customers to invest in agricultural projects in return for harvest from the field. Decentralized architecture and smart contract is the core foundation for achieving this platform since smart contract maintains trust among two involved parties and triggers unbiased payouts based on certain parameters. The success of this model can lead to developments in crowdfunding platforms, supply chain management systems as well as other forms of insurance.

**Keywords**: *Blockchain; Supply Chain; Ethereum; Crowd farming; Smart-contracts; Decentralized-Application*

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| **DApp** | Decentralized application |
| **HTML** | Hypertext Markup Language |
| **CSS** | Cascading Style Sheet |
| **JS** | JavaScript |
| **P2P** | Peer-to-peer |
| **UML** | Unified Modeling Language |
| **NPM** | Node Package Manager |

# CHAPTER 1: INTRODUCTION

## 1.1. Overview

Blockchain technology is one of the most trending topics of today's software world. Initially, it was only used to build cryptocurrencies. However, recent studies have focused on the use of blockchain in various other fields. Advanced security and transparency provided by the blockchain technology can transform the future of security. Thus, we can utilize it in areas that require privacy as well as transparency. The distributed structure of blockchain can be used to store vital information such as legal certificates, bank account books, government documents, etc. Furthermore, it can also be used to conduct events such as election campaigns. Another increasingly popular concept is the use of smart contracts implemented by Ethereum. A smart contract is a self-executing contract with the terms of the agreement between buyer and seller being directly written into lines of code. The code and the agreements contained therein exist across a distributed, decentralized blockchain network. The code controls the execution, and transactions are trackable and irreversible.

## 1.2. Background and Motivation

The main aim of developing this project is to build a platform for farmers to secure their investment and revenue well before the harvest which is governed and maintained using smart contracts between the farmers and the users. Crowd farming, though similar to crowdfunding, is a relatively new concept in which customers pay money to the farmer in the early stages of planting the desired crop as a seed investment and later get return in the form of harvest from the field. So, customers become the investors of the project and farmers are able to initiate large-scale farming projects with low risk and high profit since the customers are already on board with the farmers in the farming process right from the start. This also ensures a sense of accountability of both farmers and customers towards the project.

Along with that, the application also aims to build a reliable supply chain data which is

accessible online. Leading a healthy lifestyle and consuming healthy products are customer's rights. Thus, it is essential to have transparent data trusted by the people. The proof of work mechanism practiced in the blockchain technology can be used to ensure transparency while maintaining the timestamps. Blockchain stores information across various P2P network, so there is no single point of data loss. Thus, timestamp manipulation can be eradicated. Furthermore, decentralization ensures the security of the entire data stored in all devices even when one system fails.

## 1.3. Problem Statement

Nepal still has a long way to go in terms of providing farmers with a proper market and opportunities to sell their products. Lots of fruits and vegetables are imported from foreign countries like India but the massive agricultural harvests of Nepalese farmers remain unsold, and a lot of time wasted. For paddy, the minimum support price fixed by the Government in 2078 is Rs 29.02 per kg but the farmers are offered as low as Rs 12 per kg in certain parts of the far western region of Nepal [3]. Poor management within the government bodies and a bunch of middlemen trying to rip off profit margins from farmers has caused farmers to only hope for their investment to be compensated rather than making any profit. A major reason for this issue is also due to the centralized development and a gap that exists between the farmers and the customers.

Prices of vegetables and fruits keep on hiking on a daily basis and the ones sold as 100% organic have no authentic proof of organicness except for a label. Along with that, there are various concerns regarding the excessive use of pesticides and fertilizers by farmers with no proper regulations and verification from the concerned department.

This project aims to build a crowdfarming platform with a supply chain that allows farmers to run farming campaigns and customers can participate in the projects by contributing a certain amount and then waiting for the return of their investments as fruits and vegetables, fresh from

the orchid. Also, the supply chain provides the status of the project at each stage to the user and the user can always check the timestamps of the various events that are secured in the blockchain network.

## 1.4. Objectives

### 1.4.1 General Objective

To implement a decentralized application to provide a platform to facilitate crowd farming.

### 1.4.2. Specific Objectives

a) To prevent middlemen from influencing crop prices and directly connect farmers and customers.
b) To encourage large-scale agricultural projects with crowd farming.
c) To prevent any form of control over the fund collected and decentralize the power to allow withdrawal of money to investors.
d) Maintain transparency of project's status and crop details with the supply chain.

## 1.5. Scope and Limitation

The project falls under the domain of Decentralized Application, which is a field of Blockchain that uses smart contracts to secure agreements between two involved parties and ensure transparency and trust.

This project focuses on how to make use of smart contracts to enable crowd farming. Enabling crowd farming is a challenge, considering the amount of trust that needs to be developed for multiple people to invest in a project and ensure that the misuse of the investment is not possible.

The scope of this project is limitless, as any market functions based on trust and transparency among the seller and the buyer. Citizens demand transparency of the budget and work done by

the government from the collected tax. This too can be achieved with the right use of decentralized application.

One major limitation of this project is the stability of this technology. Smart contracts are still in a developing stage and a lot of improvements are still being done which might make it seem immature to be brought into practice immediately.

## 1.6. Development Methodology

In order to carry out this project, an Agile Methodology (Scrum) of the SDLC has been used. The requirements of the system were broken down into several standalone modules such as smart contract development and deployment, connecting and interacting with the contract. They were further divided into milestones that were to be met in order to reach the goal. For smart contracts, functions required were developed, deployed in the Local Blockchain Network and tested to ensure the smooth functioning of all the functionalities before the integration with the main platform.

## 1.7. Outline

The report is organized as follows:

**Preliminary Section:** This section contains the title page, abstract, table of contents, list of figures, and list of tables.

**Introduction Section:** In this section, the overview of the project, the background and motivation of the project, problem statement, its objectives and scope are discussed.

**Literature Review Section:** This section includes description, summary and critical evaluation of all the research papers studied to build a foundation of knowledge required for this project.

**Requirement and Feasibility Analysis Section:** Requirement analysis, and feasibility analysis make the bulk of this section.

**System Design Section:** The section consists of description of data used, algorithms implemented and the system design as well.

**Development Methodology Section:** This describes the software development lifecycle followed to build this system.

**Implementation and Evaluation Section:** The section comprises the tools and technologies used to build the system, description of implementation, and results obtained after system testing.

**Conclusion and Recommendation Section:** The section is composed of the final findings and the recommendations that can be worked on to improve the project.

# CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW

## 2.1. Background Study

Blockchain is a relatively new technology that has gained mass interest after the popularity of Bitcoin. Bitcoin is a cryptocurrency which is implemented on top of blockchain. It's just one of the applications of blockchain however, its potential is limitless due to its decentralized structure. In traditional architecture our applications, files and data are stored in a central repository in a server. At best, the files are stored redundantly across multiple servers to achieve decentralization but the power to access and modify the data remains centralized. Blockchain fills the gap and makes the data truly decentralized in sense that the files are stored as records in a ledger across millions of users (called nodes) in the network and every node in the network checks for unauthorized changes in the stored data making it impossible to make modifications once the data is published in the network.

Summarizing from a couple of research papers and project documentation, the main features required by an electronic crowd farming system are the control over the funds collected, authority to block unnecessary withdrawals, and transparency of funds being collected. All of these requirements are fulfilled by a decentralized application.

## 2.2. Literature Review

Summarizing from a couple of research papers and project documentation, the main feature required by an electronic crowd farming and supply chain system is the control over the fund collected by the users, authority to block unnecessary withdraw, transparency of timestamps and states and allowing only the concerned farmer to update the status of any issued crop. Similarly, this system also supports the aforementioned features. Furthermore, a feature that

was not implemented in any of the projects as farmer (seed buyers) authenticity. This system implements a mechanism to identify a project using the project's token.

**Crowdfarming.com**, a company in Europe that implements crowd farming, is a project that is comparable to the one that we have picked. However, it is totally database-driven, while our the project is built on the blockchain. Customers in our projects will have more control and transparency over the funds provided as a result of the use of blockchain.

**E-farm**, a Nepalese startup with a similar mission to ours, namely, connecting farmers and consumers. However, their approach is more e-commerce-oriented, and they lack a way for determining whether a product is organic and local. Because our project has a timestamping and status feature, everything is recorded from the status of "planted" to "shipped" for any given crop, we can give a solid solution to this problem.

## 2.3. Current System

Currently, there are few only groups of people coming together and running a crowd farming campaign. But not much implementation of the concept can be observed. Talking about the supply chain aspect of the project, there are only paper works being used in terms of Nepal.

## 2.4. The problem with Current System

Crowdfarming only based on the trust can be risky and chances of fraud are high if the collected funds are not utilized by the farmer properly or unnecessary withdrawals are made. Along with that, a supply chain based on paper works doesn't provide a reliable source of information and can be easily tampered to alter the data.

# CHAPTER 3: SYSTEM ANALYSIS

## 3.1. Requirement Analysis

After analyzing the existing system and the necessity to create a secure fund raising platform, the following features were added to the system:

- Farmers shall create a crowd farming project with targeted amount and minimum investment amount.
- A farmer cannot create more than one crowd farming project at a time.
- Only the farmer who created the project can make a withdraw request
- Only the investors associated with a crowd farming project can approve withdraw requests
- Withdrawal requested from a farmer will only be released to the farmer's account if 51% of the investors approve the request.

### 3.1.1. Functional Requirement

The functional requirements of the project are:

- The farmer shall create a project that initiates a contract
- Investor shall send money as fund to the contract
- Investor shall contribute to multiple crowd farming projects
- The farmer shall request multiple withdraws
- Investor shall approve a withdraw request
- The farmer shall be allowed to enter his/her id code

### 3.1.2. Non-Functional Requirement

The non-functional requirements of the project are:

- Credentials provided (input) by the farmers must be registered beforehand as farmers.
- Only verified farmer must initiate a crowdfarming project
- Only verified investors can invest in a crowd farming project
- Withdraw request must be approved only by the investors
- 51% of the investors must approve the withdraw request for the farmer to withdraw the amount
- The user's system must have a stable Internet connection.

## 3.2. Feasibility Analysis

After the requirement collection, the feasibility analysis was carried to check the feasibility of the project.

### 3.2.1. Technical Feasibility

This project is based on Ethereum Smart Contract that is accessed via Metamask which is a wallet provider available as a browser extension. It is a web based application whose user interface is developed in React.js and backend architecture is developed in Node.js. Hence all the tools, modules and its dependency is needed to build the system are open source and easy to understand. The community of the technology being used is prominent. So, the project was technically feasible.

### 3.2.2. Operational Feasibility

The operational feasibility depends upon the working of the system. Farmers create a project and general users can invest money in the created project.. This project is implemented as a web-based application due to which it can be accessed by any web browser. The cost, time and

resources required to run this system are very minimal. Hence this project is operationally feasible.

### 3.2.3. Economic Feasibility

This software and resource used were open source marking it as economically feasible. The blockchain test net was set up in the local machine with dummy ethers to enable transactions and for the research purpose internet connection was used. The project is web based and requires installation of metamask wallet provider which is freely available as a browser extension. Hence no direct expenses were required for the successful completion and building of the project.

### 3.2.4. Schedule Feasibility

The schedule feasibility analysis is carried out using the CPM method. All the critical tasks were identified and figured out ensuring that all the activities are performed as scheduled .Hence project was completed within the desired time frame. From the above network diagram, the critical path is A, B D, F and H. The CPM analysis was carried out as follows:

**Table 1: Project Task Schedule**

| **Tasks** | **Duration (days)** | **Predecessors** |
|---|---|---|
| Research on Blockchain and Smart Contract (A) | 5 | |
| Requirement Collection (B) | 4 | A |
| Developing Smart Contract (C) | 5 | A |
| Backend Development (D) | 10 | B |
| UI Design (E) | 5 | C |
| System Integration (F) | 4 | C |
| System Testing (G) | 3 | D, E |
| Documentation (H) | 3 | F, G |

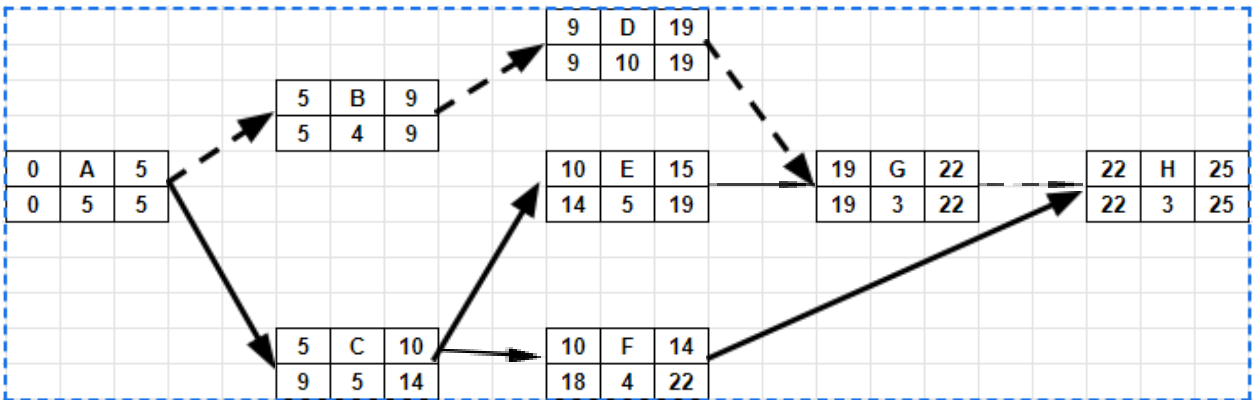



**Figure 1: Network Diagram to Identify Critical Path**

Index
ES : Early Start
EF : Early Finish
LS : Late Start
LF : Late Finish

Figure 2 shows the network diagram for finding the critical path of the project. As shown above, critical tasks are (A) Research on Smart Contract, (B) Data Collection, (D) Backend Development, (G) System Testing and (H) Documentation. The total duration of the critical path is 85 days which is within the deadline range. Hence, this project is feasible in terms of schedule feasibility.

# CHAPTER 4: SYSTEM DESIGN

## 4.1. Design

The system architecture is elaborated using Unified Modeling Language (UML) diagrams listed in the table below:

### 4.1.1. Flow Diagram of Project

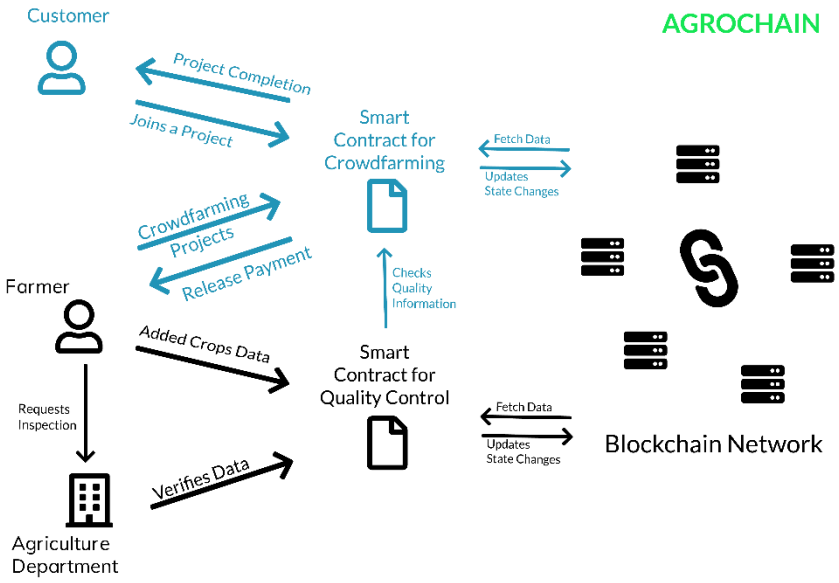



**Figure 2: Flow diagram of project**

### 4.1.2. Use Case Diagram

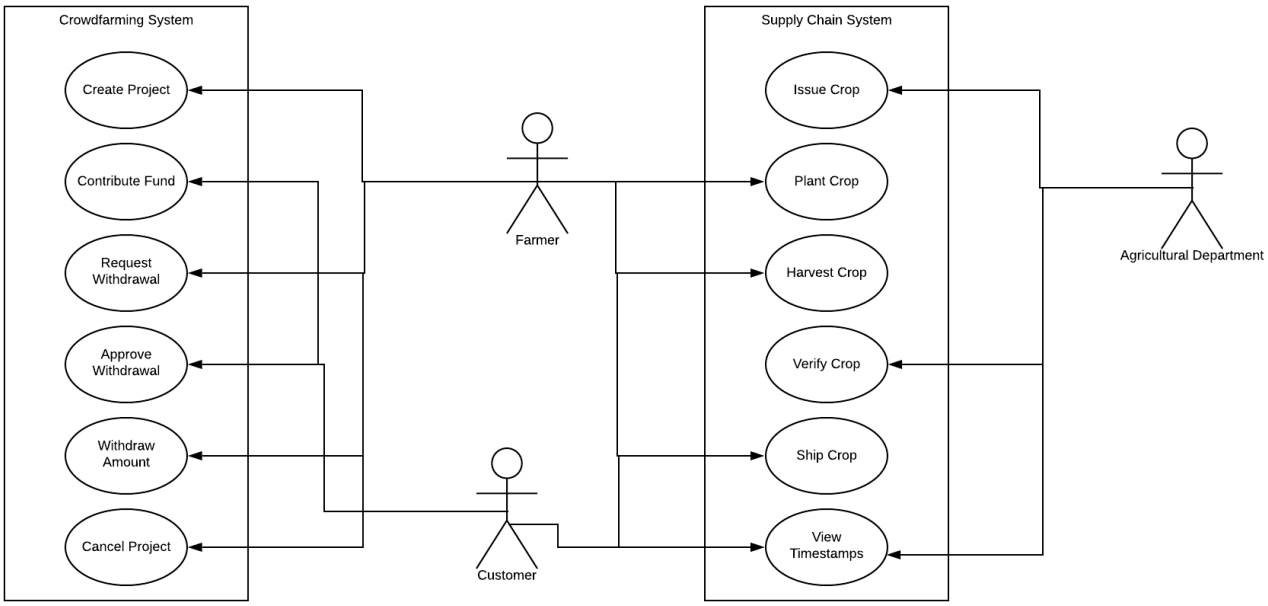



**Figure 3: Use case diagram of project**

### 4.1.3. Sequence Diagram

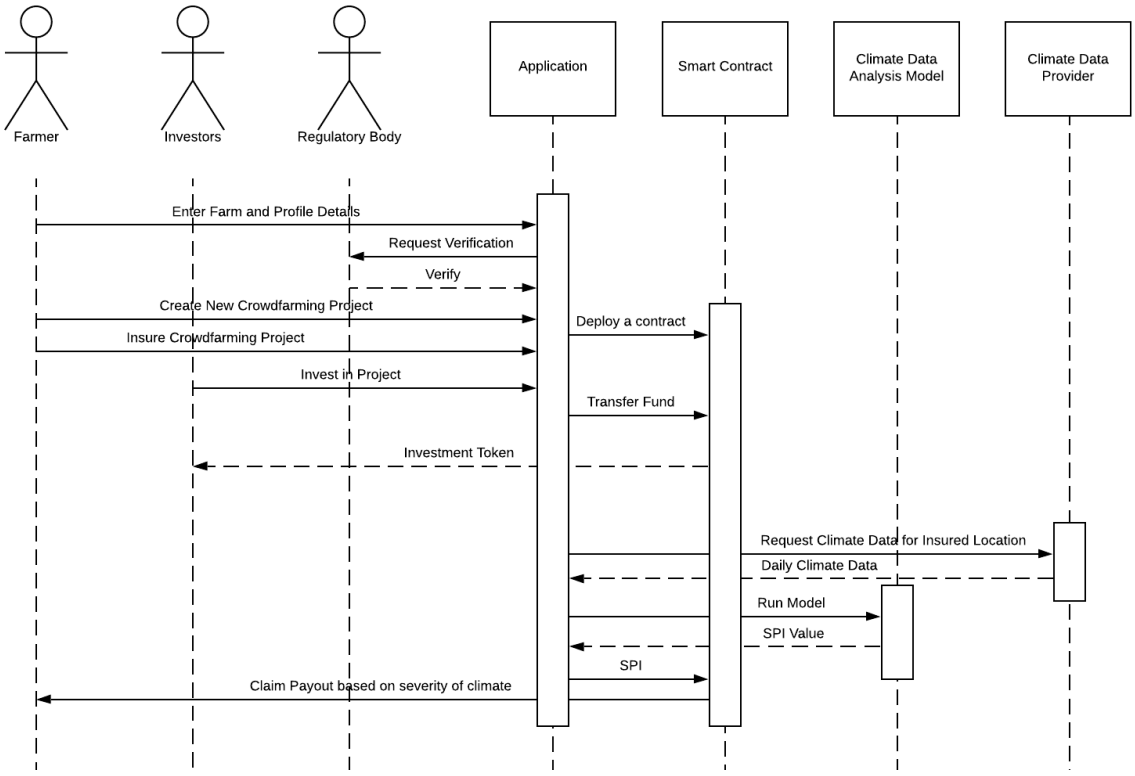



**Figure 4: Sequence diagram of project**

## 4.2. Algorithm Details

### 4.2.1. SHA-256 Algorithm

SHA 256 is a part of the SHA 2 family of algorithms, where SHA stands for Secure Hash Algorithm. Published in 2001, it was a joint effort between the NSA and NIST to introduce a successor to the SHA 1 family, which was slowly losing strength against brute force attacks.

**Characteristics of the SHA-256 Algorithm**

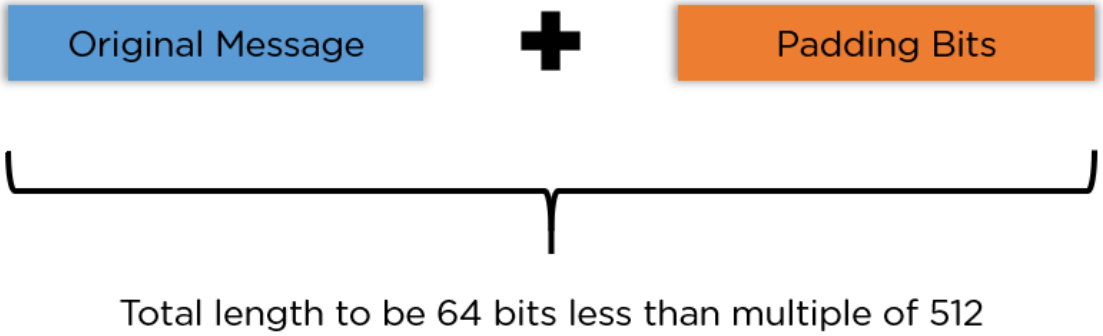
- **Message Length:** The length of the cleartext should be less than 264 bits. The size needs to be in the comparison area to keep the digest as random as possible.
- **Digest Length:** The length of the hash digest should be 256 bits in SHA 256 algorithm, 512 bits in SHA-512, and so on. Bigger digests usually suggest significantly more calculations at the cost of speed and space.
- **Irreversible:** By design, all hash functions such as the SHA 256 are irreversible. You should neither get a plaintext when you have the digest beforehand nor should the digest provide its original value when you pass it through the hash function again.

**Steps in SHA-256 Algorithm**

You can divide the complete process into five different segments, as mentioned below:

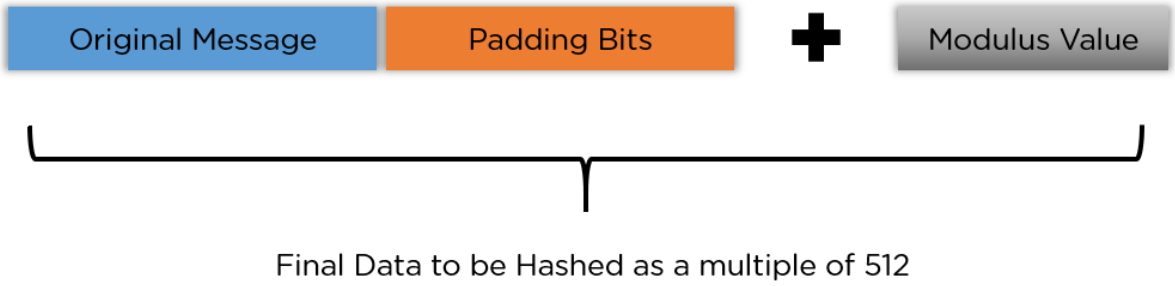
**Padding Bits**

It adds some extra bits to the message, such that the length is exactly 64 bits short of a multiple of 512. During the addition, the first bit should be one, and the rest of it should be filled with zeroes.




**Padding Length**

You can add 64 bits of data now to make the final plaintext a multiple of 512. You can calculate these 64 bits of characters by applying the modulus to your original cleartext without the padding.




**Initializing the Buffers:**

You need to initialize the default values for eight buffers to be used in the rounds as follows:

a = 0x6a09e667

b = 0xbb67ae85

c = 0x3c6ef372

d = 0xa54ff53a

e = 0x510e527f

f = 0x9b05688c

g = 0x1f83d9ab

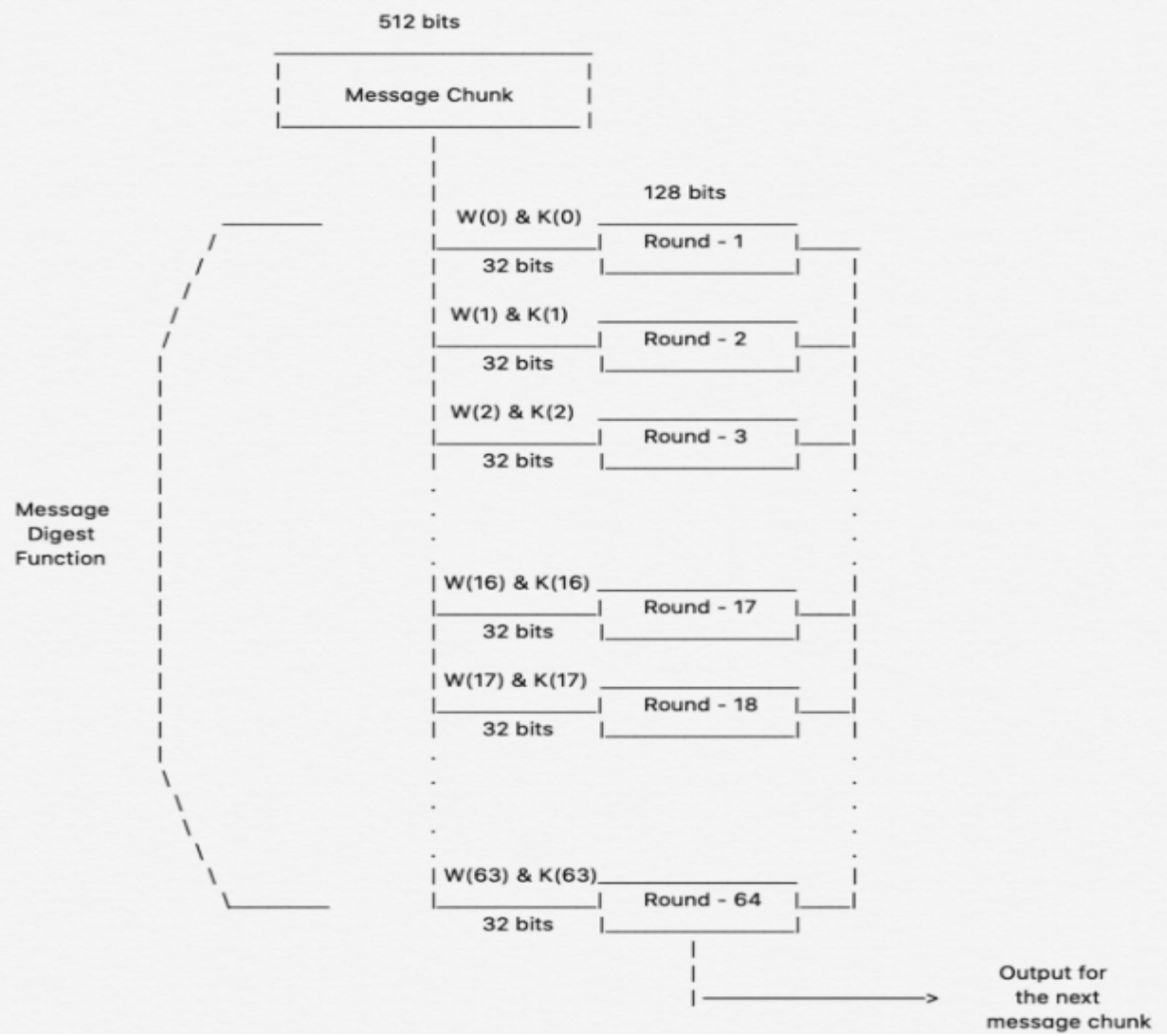
h = 0x5be0cd19

You also need to store 64 different keys in an array, ranging from K[0] to K[63]. They are initialized as follows:

```
k[0..63] :=
   0x428a2f98, 0x71374491, 0xb5c0fbcf, 0xe9b5dba5, 0x3956c25b, 0x59f111f1, 0x923f82a4, 0xab1c5ed5,
   0xd807aa98, 0x12835b01, 0x243185be, 0x550c7dc3, 0x72be5d74, 0x80deb1fe, 0x9bdc06a7, 0xc19bf174,
   0xe49b69c1, 0xefbe4786, 0x0fc19dc6, 0x240ca1cc, 0x2de92c6f, 0x4a7484aa, 0x5cb0a9dc, 0x76f988da,
   0x983e5152, 0xa831c66d, 0xb00327c8, 0xbf597fc7, 0xc6e00bf3, 0xd5a79147, 0x06ca6351, 0x14292967,
   0x27b70a85, 0x2e1b2138, 0x4d2c6dfc, 0x53380d13, 0x650a7354, 0x766a0abb, 0x81c2c92e, 0x92722c85,
   0xa2bfe8a1, 0xa81a664b, 0xc24b8b70, 0xc76c51a3, 0xd192e819, 0xd6990624, 0xf40e3585, 0x106aa070,
   0x19a4c116, 0x1e376c08, 0x2748774c, 0x34b0bcb5, 0x391c0cb3, 0x4ed8aa4a, 0x5b9cca4f, 0x682e6ff3,
   0x748f82ee, 0x78a5636f, 0x84c87814, 0x8cc70208, 0x90befffa, 0xa4506ceb, 0xbef9a3f7, 0xc67178f2
```

**Compression Functions**

The entire message gets broken down into multiple blocks of 512 bits each. It puts each block through 64 rounds of operation, with the output of each block serving as the input for the following block. The entire process is as follows:




While the value of K[i] in all those rounds is pre-initialized, W[i] is another input that is calculated individually for each block, depending on the number of iterations being processed at the moment.

**Output**

With each iteration, the final output of the block serves as the input for the next block. The entire cycle keeps repeating until you reach the last 512-bit block, and you then consider its output the final hash digest. This digest will be of the length 256-bit, as per the name of this algorithm.

# CHAPTER 5: IMPLEMENTATION AND TESTING

## 5.1. Implementation

Since the system is locally deployed, firstly, local blockchain is started using the terminal. The constructor defined in the smart contract is executed once the blockchain is started. The constructor is defined to initialize the list of projects and some dummy users.

The system starts when a user hits its URL. The user is then required to enter their Metamask ID and password. If the user is not registered as a farmer then they need to register in the system as a farmer. The farmer needs to be verified by the authorized department. The farmer is then allowed to create a project, select crops and plant them. Once a project is initiated, investors will be able to contribute if the project interests them.

General users can register as crowd farmers and select projects initiated by farmers and invest a minimum investment amount. The money is entirely in control of the smart contract and no withdrawal can be made without the investor's consent. Farmers can request for withdrawal and once 51% of the investors approve the request then the request can be finalized and withdrawn. The project can be canceled by the farmer at any time and the remaining fund in the contract gets distributed back to the investors.

### 5.1.1. Tools Used

The following dependencies installed to setup the system used to build the e-voting application:

**NPM:** Node Package Manager to install and update different JavaScript modules used in the project.

**Truffle Framework:** provides tools for writing and testing smart contracts using Solidity programming language and for deploying smart contracts to the blockchain.

**Ganache:** local n-memory blockchain used for development purpose

**Metamask extension for Google Chrome:** connects the browser to local Ethereum and interact with the smart contract

**HTML/CSS:** used for frontend designing

**JavaScript:** used to handle events in the frontend and trigger functions in the smart contract for backend

**React.js:** used to create the frontend architecture

**Node.js:** used to create the backend architecture

### 5.1.2. Implementation Details of Modules

**File: Contract.js**

For the interaction with the smart contract, web3.js is used. Web3 instances are instantiated and different contract methods are executed using this interface. Important functions in contract.js are:

**loadWeb3:** Asynchronous function to authenticate use using Metamask ID and fetch account details and balance.

## 5.2. Testing

### 5.2.1. Test Cases for Unit Testing

**Table 2: Test Cases for Unit Testing of Server Application**

| Test Case # | Test Case Description | Test Data | Expected Result | Actual Result | Pass/Fail |
|---|---|---|---|---|---|
| TU01 | Check Customer Login with valid Data<br>Go to site /login<br>Enter UserId<br>Enter Password<br>Click Login | Userid = 0x7fb867f10E848E19381773D2b07e76ac9f5E261E<br>Password = admin | User should Login into an application | As Expected | Pass |
| TU02 | Check Customer Login with invalid | Userid = 0x7fb867f10E848E19 | User should be redirected to the | As Expected | Pass |

| | | | | | |
|---|---|---|---|---|---|
| | Data<br>Go to site /login<br>Enter UserId<br>Enter Password<br>Click Submit | 381773D2b07e76ac9f5E261E<br>Password = password | login page with invalid email/password error | | |
| TU03 | Create Crowdfarming Project<br>Go to Dashboard > Create Project<br>Enter project details<br>Click Submit | Project Name : Avocado<br>Targeted Amount : 500000<br>Minimum Amount : 10000 | User should be prompted by metamask to deploy a contract | As Expected | Pass |
| | | | | | |
| | | | | | |

### 5.2.2. Test Cases for System Testing

**Table 3: Test Cases for System Testing of Server Side and Blockchain Application**

| Test Case # | Test Case Description | Test Data | Expected Result | Actual Result | Pass/Fail |
|---|---|---|---|---|---|
| TU01 | Withdrawal of crowdfunded money | Withdrawal request : For fertilizers | Investors should receive notification to approve the request | As Expected | Pass |
| TU02 | Approval of Withdrawal Request | | Withdraw button activated in farmer's dashboard | As Expected | Pass |

## 5.3. Result Analysis

The system could only be tested by adding 10 users. Proper tests could have been carried out if the local blockchain network allowed creating more accounts. Nevertheless, the system performed with the same accuracy and speed for one user as well as 10 users. The system successfully validated all the users' IDs and also checked whether they should be allowed to perform restricted transactions or not.

# CHAPTER 6: CONCLUSION AND FUTURE RECOMMENDATION

## 6.1. Conclusion

The idea of implementing crowdfarming is to create a secure farming opportunity for the farmers by pre-collecting the money through the campaign and the investors too can be ensured about fresh fruits and vegetables direct from the farm to reach their door steps.

Similarly, the idea of replacing the traditional supply chain with an electronic agricultural supply chain is to build reliable supply chain data which is accessible online. Leading a healthy lifestyle and consuming healthy products are customer's rights. Thus, it is essential to have transparent data trusted by the people. The proof of work mechanism practiced in the blockchain technology can be used to ensure transparency while maintaining the timestamps. Blockchain stores information across various P2P networks, so there is no single point of data loss. Thus, timestamp manipulation can be eradicated. Furthermore, decentralization ensures the security of the entire data stored in all devices even when one system fails.

Some limitations of the system incurred due to multiple factors are listed below:

- Functionality of Metamask extension could not be overwritten. So, confirmation of transactions could not be customized.
- Latest version of web3 package lacked functionality to set coinbase (message sender's address). Thus, the Metamask functionality had to be used to accept farmer's id as input.
- Transparency of timestamps could not be ensured considering psychological facts.
- Use of Ganache emulator barred generating new accounts. So, adding, editing or deleting farmers could not be implemented using a user interface.

## 6.2. Future Recommendation

The system can be further developed to include our own private blockchain network to enhance the performance since private blockchains are faster to process a high number of transactions & create personal tokens to facilitate transactions. Also we can make the system multiple lingual so that people from different backgrounds find it easy to use.

# REFERENCES

[1] Unknown, "Unseasonal rainfall caused a loss of Rs 7.22 billion to Nepal farmers" Online Khabar, October 22, 2021. [Online].

Available: https://www.onlinekhabar.com [Accessed December 10, 2021].

[2] https://builtin.com
[3] https://searchcompliance.techtarget.com
[4] https://www.guru99.com
[5] https://www.crowdfarming.com/
[6] https://www.efarm.live/