**AGRICULTURE DOCS CHAIN**

**A PROJECT REPORT**

***Submitted by***

|  |  |
| --- | --- |
| KALPANA A | **622620104008** |
| NANDHINI J | **622620104015** |
|  |  |
|  |  |
|  |  |

# TEAM ID: NM2023TMID11506

In partial full fillment for the award of the degree

***Of***

BACHELOR OF ENGINEERING

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**SHREENIVASA ENGINEERING COLLEGE**

**BOMMIDI ,DHARMAPURI-636807**

TABLE OF CONTENT

1. INTRODUCTION
   1. Project Overview
   2. Purpose
2. LITERATURE SURVEY
   1. Existing problem
   2. References
   3. Problem Statement Definition
3. IDEATION & PROPOSED SOLUTION
   1. Empathy Map Canvas
   2. Ideation & Brainstorming
4. REQUIREMENT ANALYSIS
   1. Functional requirement
   2. Non-Functional requirements
5. PROJECT DESIGN
   1. Data Flow Diagrams & User Stories
   2. Solution Architecture
6. PROJECT PLANNING
   1. Technical Architecture
7. CODING & SOLUTIONING
   1. Feature
8. PERFORMANCE TESTING
9. RESULTS
   1. Output Screenshots
10. ADVANTAGES & DISADVANTAGES

10.1Advantages

10.2 Disadvantages

1. **CONCLUSION**
2. FUTURE SCOPE
3. **APPENDIX** 
   1. Source Code
   2. GitHub & Project Demo Link

# 1.1 INTRODUCTION

**1.1 PROJECT OVERVIEW:**

Agriculture is the backbone of many economies, providing sustenance and livelihoods for billions of people across the globe. However, this vital sector faces a myriad of challenges, with one of the most pressing being the vulnerability of farmers to various risks, including crop failure, natural disasters, and market fluctuations. To mitigate these risks and safeguard the livelihoods of farmers, agricultural insurance is indispensable.

Traditionally, agricultural insurance has been marred by inefficiencies, lack of transparency, and difficulties in claim settlement. These challenges often leave farmers at a disadvantage, struggling to recover from losses. The advent of blockchain technology, with its inherent features of transparency, security, and efficiency, presents a promising solution to transform the landscape of agricultural insurance.

**1.2 PURPOSE:**

The purpose of the "Farmer Insurance Using Blockchain" project is to revolutionize the agricultural insurance industry by harnessing the potential of blockchain technology to address longstanding challenges and empower farmers. This project is driven by the following key objectives:

|  |  |  |
| --- | --- | --- |
|  |  | The primary purpose is to provide farmers with a reliable and secure insurance em that safeguards their livelihoods. By leveraging blockchain, we aim to create a transparent, trustworthy, and efficient insurance platform that ensures farmers receive timely compensation when faced with unforeseen losses, be it due to crop failure, reme weather events, or market fluctuations. |
| syst  ext |

# 2. LITERATURE SURVEY

**2.1 Existing problem:**

In the existing system of agricultural insurance, various methods and approaches are used to provide coverage to farmers. However, these systems often face several challenges and limitations. Here is an overview of the existing agricultural insurance system:

|  |  |
| --- | --- |
|  | Traditional insurance companies provide insurance coverage to farmers. These companies offer a wide range of insurance products, including crop insurance, livestock insurance, and farm equipment insurance.Farmers purchase insurance policies from these companies, pay premiums, and file claims when they suffer losses due to covered events.  Many countries have government-sponsored agricultural insurance programs designed to support farmers. |

**2.2 References:**

If looking for a reference to support your project on implementing a farmer insurance system with Ethereum smart contracts, you might consider citing a relevant

academic paper or a reputable source related to blockchain technology,

Blockchain-Based Smart Contracts to Provide Crop Insurance for Smallholder Farmers in Developing Countries (2022) by IEEE.

Sustainability | Free Full-Text | Blockchain-Based Crop Insurance: A

Decentralized Insurance System for Modernization of Indian Farmers (2023) by MDPI

BLOCKCHAIN CLIMATE RISK CROP INSURANCE (2020) by Climate

Policy Initiative

**2.3 Problem Statement Definition:**

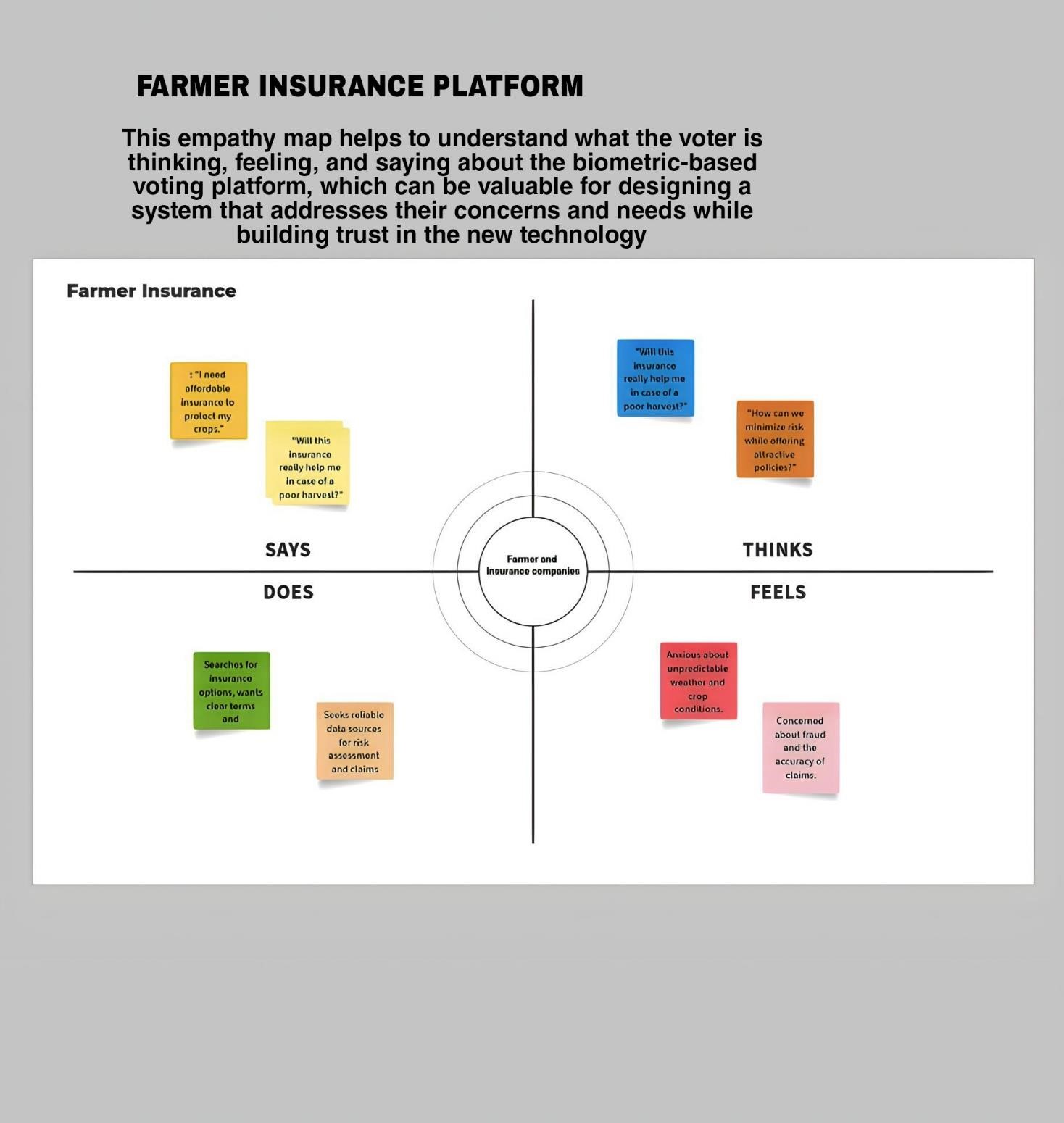
A problem statement on "Farmer Insurance Using Blockchain" defines the specific issue or challenge that your project aims to address. It provides a clear and concise description of the problem to set the stage for your project's objectives and solutions.

The current agricultural insurance system is burdened with inefficiencies, lack of transparency, and obstacles that hinder the financial security and resilience of farmers. Traditional insurance processes are marred by slow claim settlements, fraud, and high administrative costs. Smallholder farmers, in particular, face difficulties in accessing affordable and trustworthy insurance coverage, leaving them vulnerable to agricultural risks. These challenges underscore the urgent need for a transformation in the agricultural insurance sector.

The "Farmer Insurance Using Blockchain" project addresses this problem by leveraging the potential of blockchain technology. It seeks to revolutionize the insurance landscape for farmers by creating a transparent, secure, and efficient insurance system that mitigates the challenges of the current system, empowers smallholder farmers, and promotes sustainable agricultural practices.

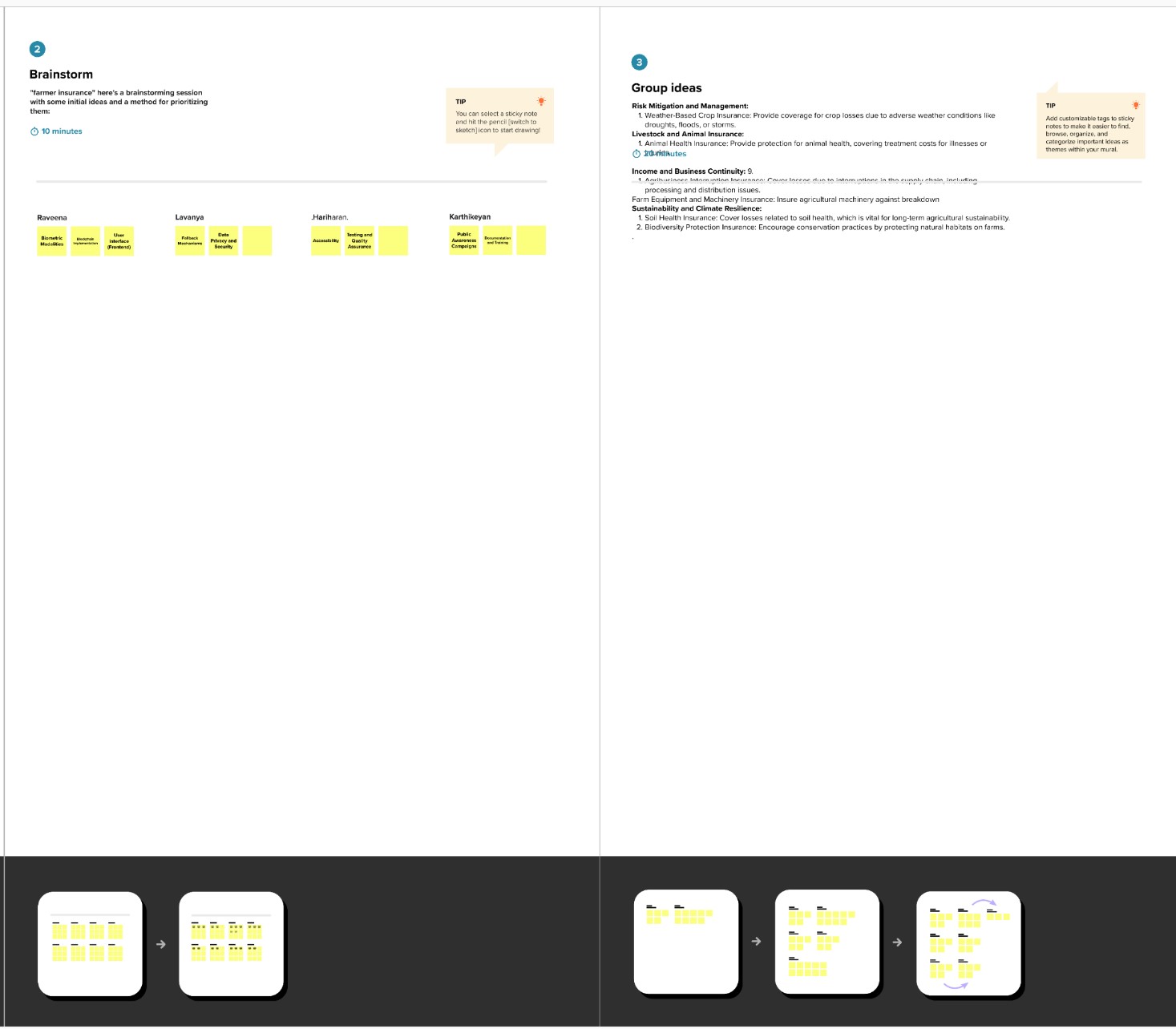
# 3. IDEATION & PROPOSED SOLUTION

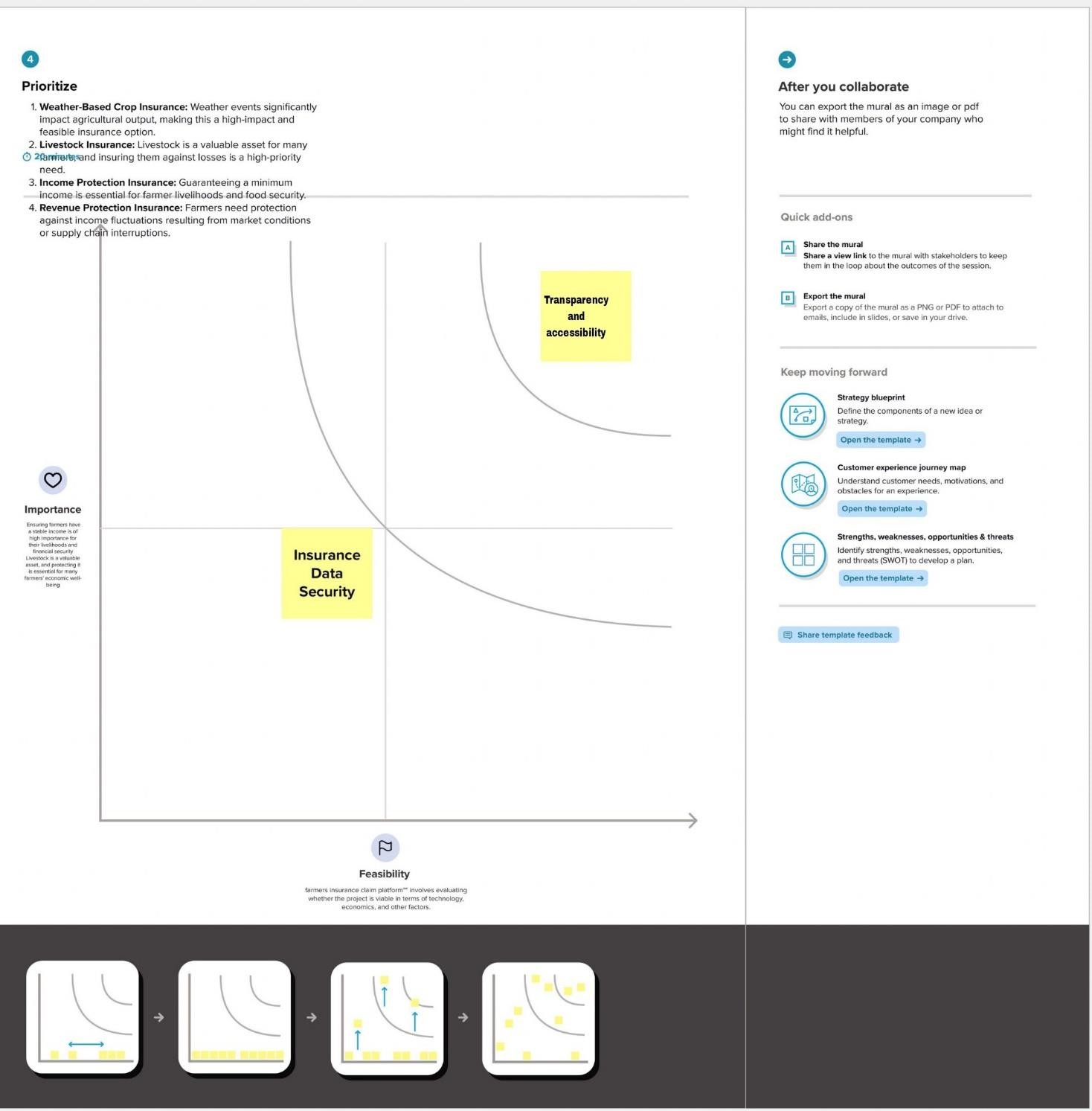
**3.1 Empathy Map Canvas:**



**3.2 Ideation & Brainstorming:**







# 4. REQUIREMENT ANALYSIS

**4.1 Functional requirement:**

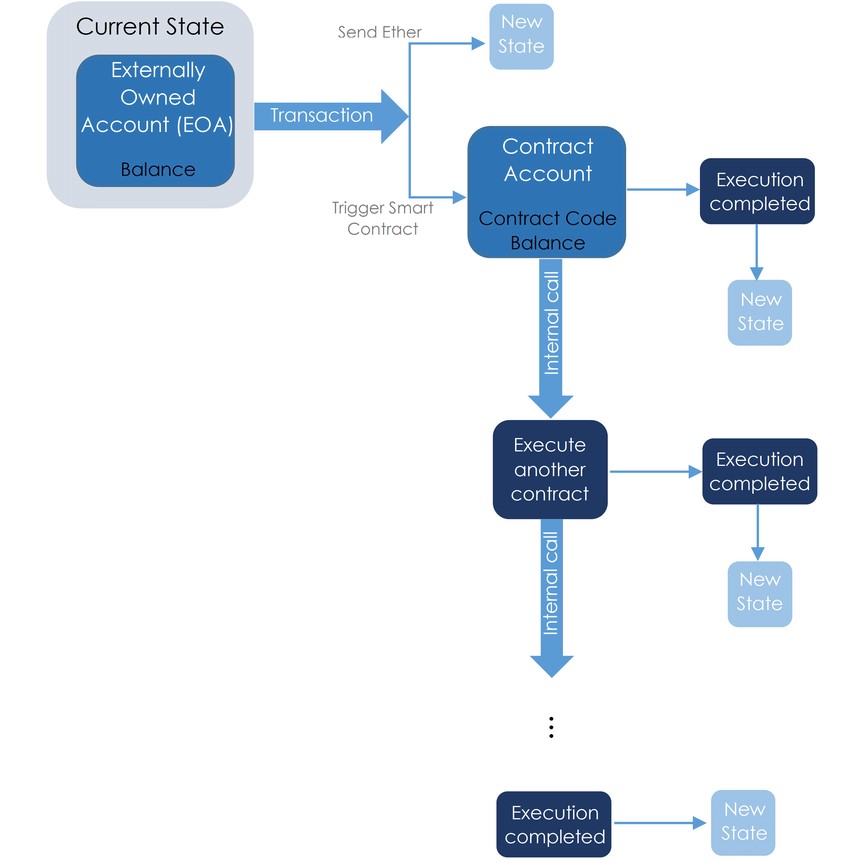
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1. User Registration and Authentication:** | | | | | thenticate themselves insurance |
| Users should be able to create accounts and au securely.Implement a user role system, differentiating between farmers, providers, and administrators. | | | | |
| **2. Policy Management:** | |  | | | |
|  | |
|  | Farmers should be able to purchase insurance policies through the platform.Define different types of policies (e.g., crop insurance, livestock insurance).Allow farmers to view and manage their active policies. | | | | | | |
| **3. Premium Calculation:** | | |  | | |
|  | | |
|  | Automatically calculate premiums based on factors such as the type of crop, coverage amount, and historical data.Display the premium amount to farmers before policy purchase. | | | | | | |
| **4. Data Input and Verification:** | | | |  | |
|  | | | |
|  | Enable farmers to input relevant data, including crop details, location, and other essential information. Implement data verification processes to ensure the accuracy of the information provided. | | | | | | |
|  | | | | | | |

**4.2. Non-Functional Requirements:**

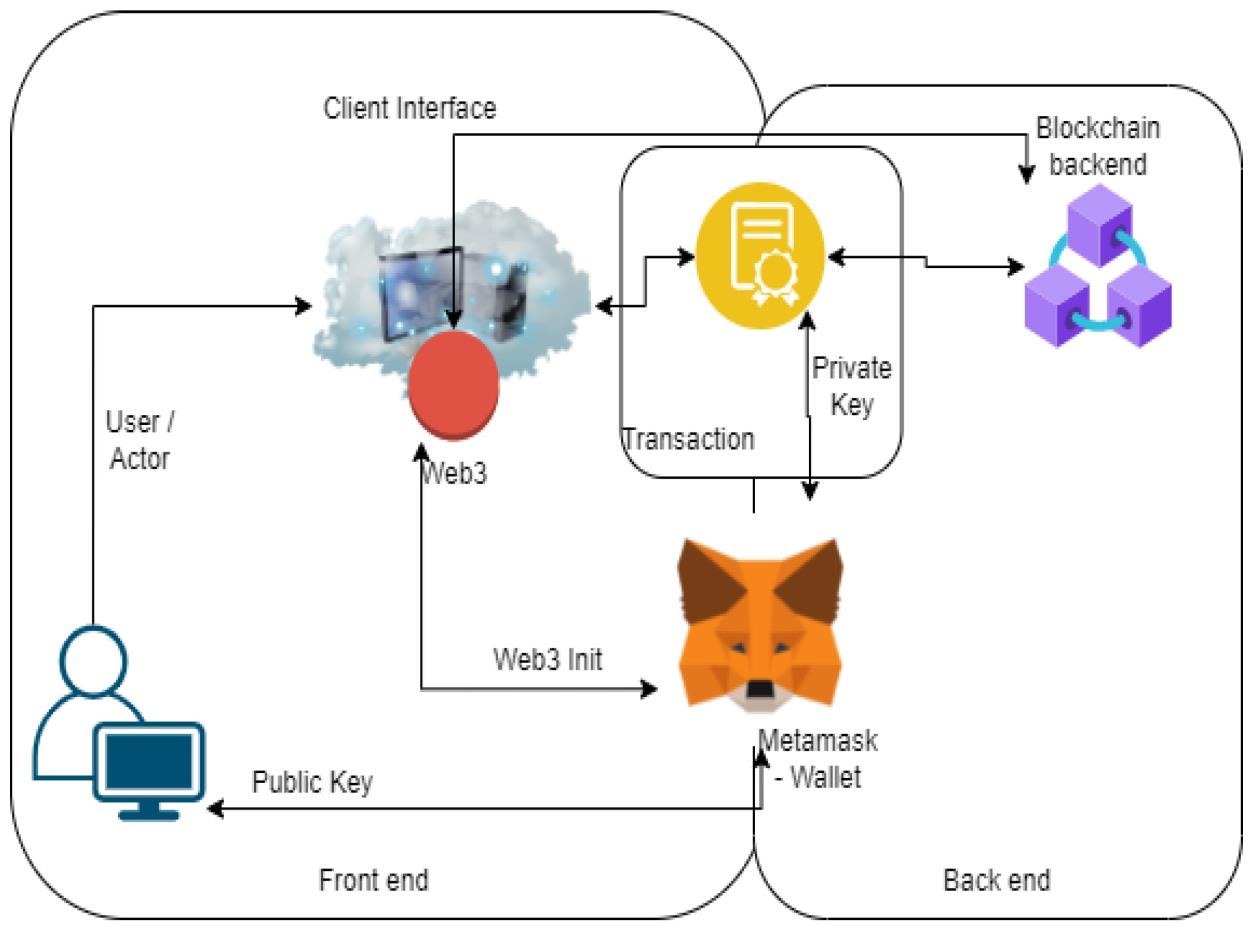
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1. Performance:** | | | | | | | The system should respond to user requests within a reasonable time frame seconds for common operations).The system should be scalable to  accommodate an increasing number of users and policies without a significant |
| (e.g., under 2  decrease in performance.  **2. Reliability:** | | | | | | |
|  |  | | | The system should be available 24/7, with planned maintenance windows  communicated to users in advance.The system should continue functioning even in the esence of hardware or software failures. | | | | | |
| pr | | |
| **3. Security:** | | | | |  | | |
|  | | | | |
|  |  | | All sensitive data, including user information and financial transactions, must be encrypted during transmission and storage. Role-based access control should be implemented to ensure that users can only access data and features relevant to their  The blockchain network should adhere to industry best practices for securing the distributed ledger. | | | | | | |
| roles. | |
| **4. Compliance:** | | | | | |  | |
|  | | | | | |
|  |  | The system should comply with relevant regulations and standards, including data protection laws and insurance industry regulation. | | | | | | | |
|  |
|  | | | | | | | | |

# 5. PROJECT DESIGN

**5.1 Data Flow Diagrams & User Stories:**

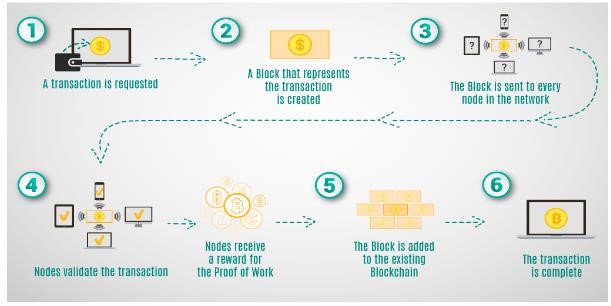


**5.2 Solution Architecture:**



# 6. PROJECT PLANNING

**6.1 Technical Architecture:**



1. **CODING & SOLUTIONING:** 
   1. **Feature:**

SPDX-License-Identifier: This specifies the license under which the code is distributed. In this case, it's using the MIT license.

Pragma solidity ^0.8.0;: This indicates that the contract is designed to work with Solidity version 0.8.0 or higher. It ensures compatibility with the specified version.

|  |  |
| --- | --- |
| **1. Development Environment Setup:** |  |
|  |

* + - Choose the programming languages and frameworks that will be used to develop the system.
    - Set up a development environment with the necessary tools, libraries, and IDEs.

|  |  |
| --- | --- |
| **2. Blockchain Network Configuration:** |  |
|  |

* + - Set up the blockchain network that will be used to store policy data, premiums, and claims.
    - Configure the blockchain's consensus mechanism, such as proof of work or proof of stake.

|  |  |
| --- | --- |
| **3. Smart Contract Development:** |  |
|  |

* + - Develop smart contracts that will govern the policy issuance, premium payment, and claims settlement processes.
    - Ensure that smart contracts are secure, tamper-proof, and efficiently written.

|  |  |
| --- | --- |
| **4. User Registration and Authentication:** |  |
|  |

* + - Implement user registration and authentication functionality.
    - Ensure secure storage and management of user credentials and roles.

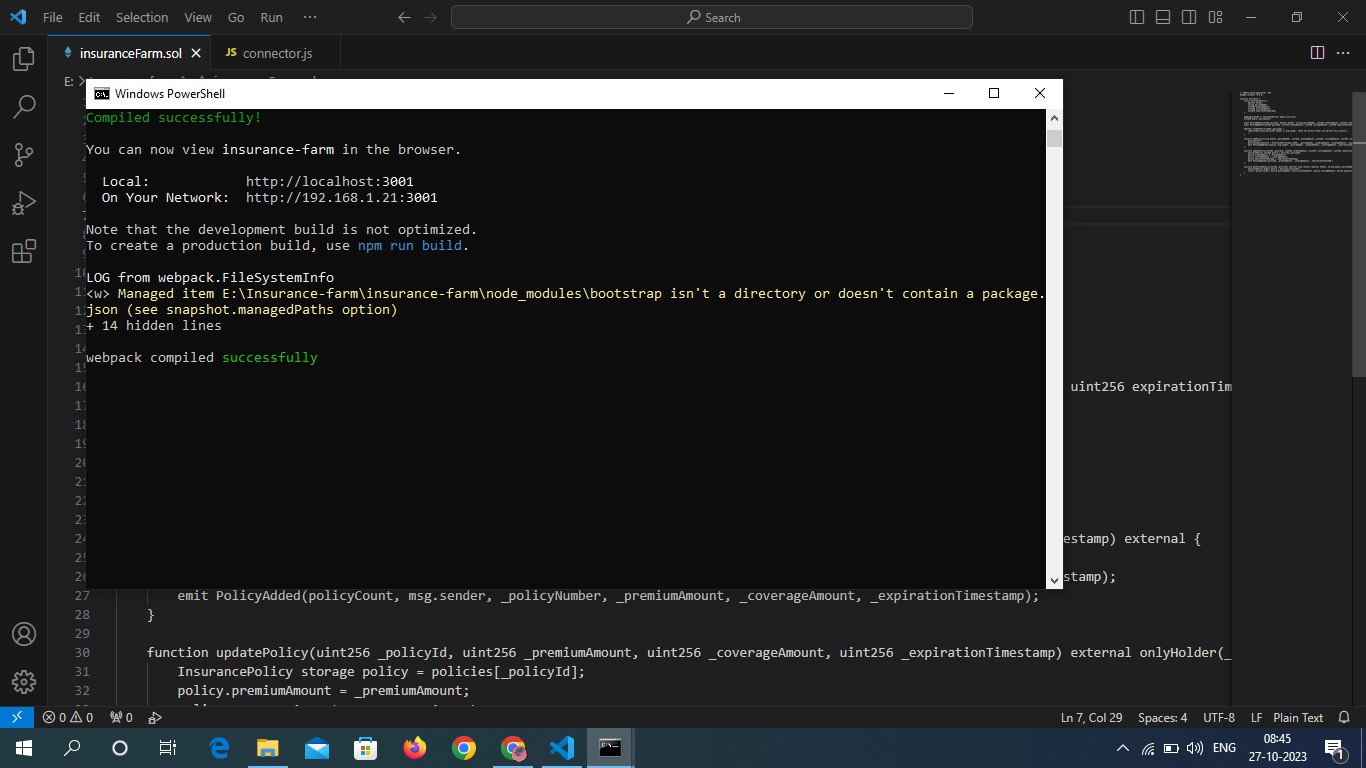
1. **PERFORMANCE TESTING:** 
   1. **Performance Metrics:**

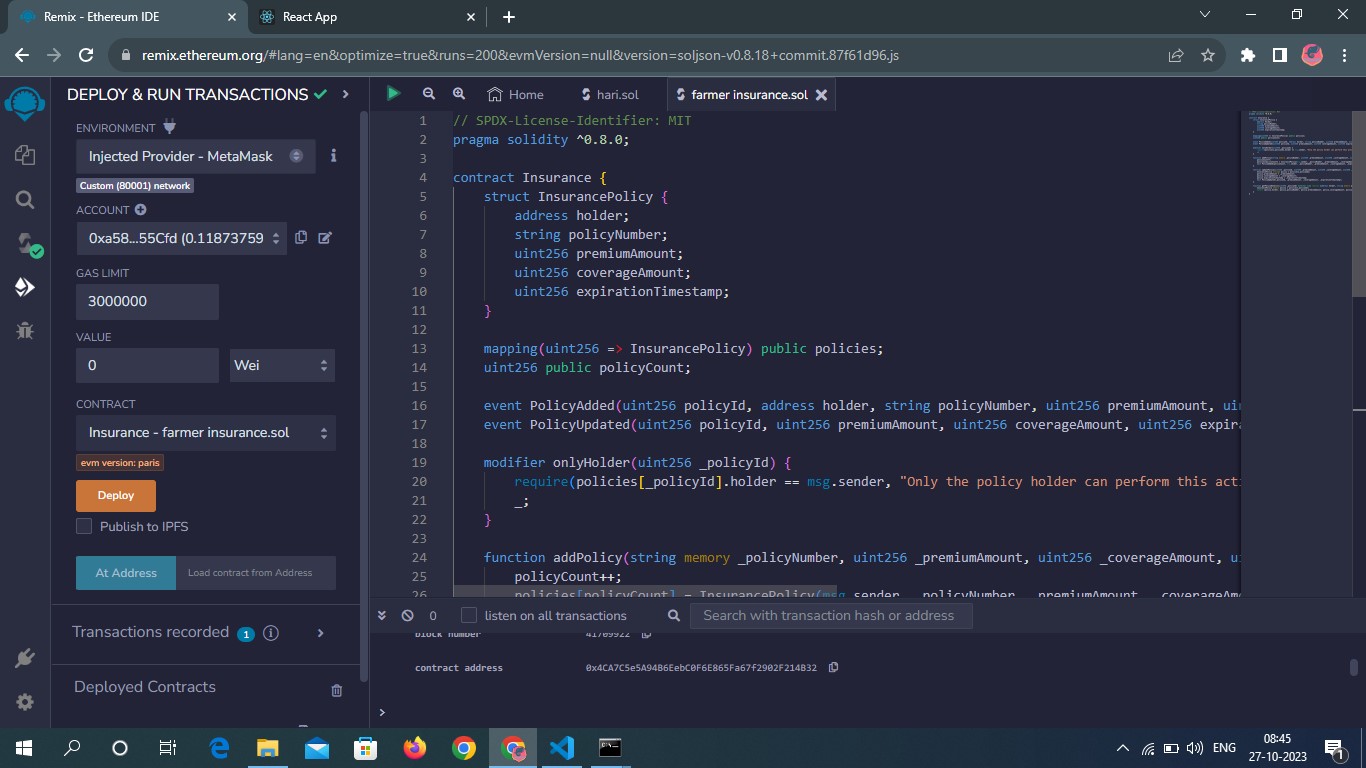
|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Parameter** | **Values** | **Screenshot** |
| 1. | Information gathering | Setup all the Prerequisite: |  |
| 2. | Extract the zip files | Open to vs code |  |

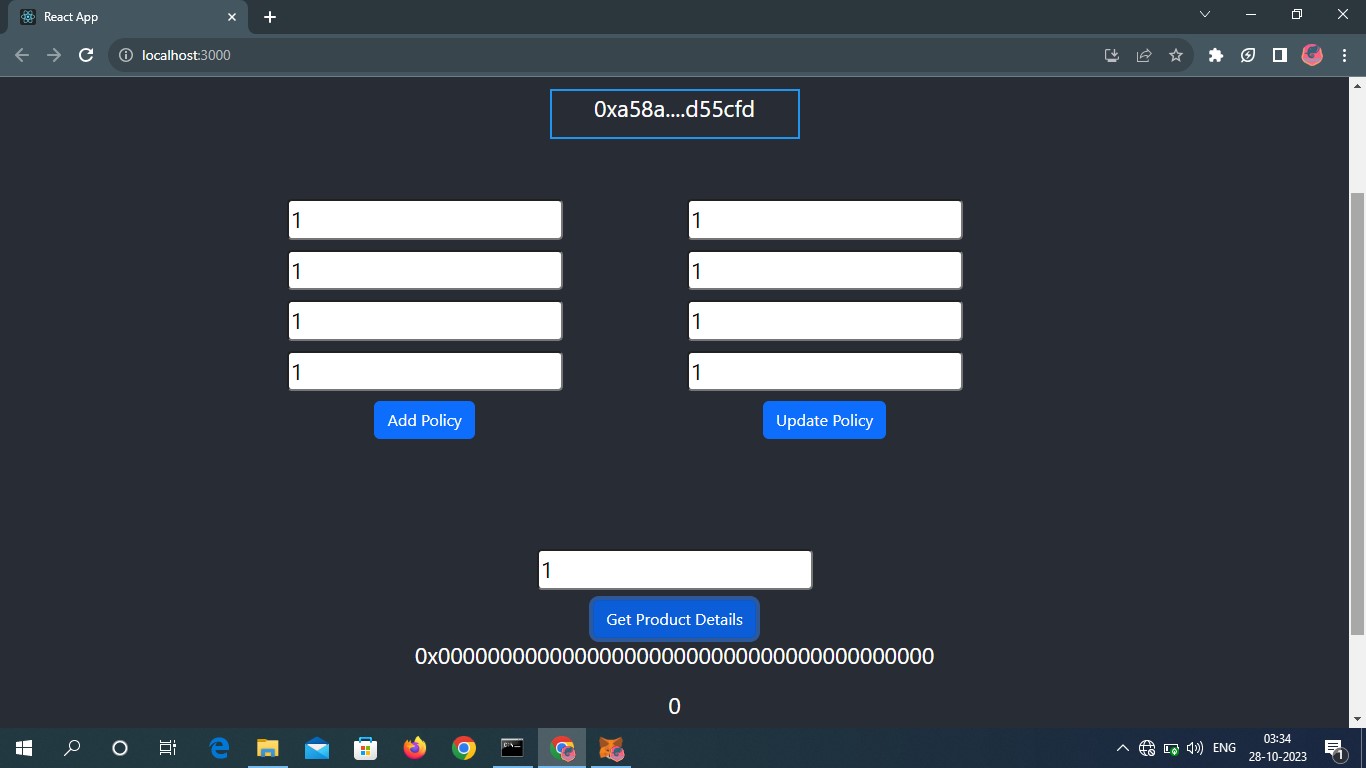
|  |  |  |  |
| --- | --- | --- | --- |
| 3. | Remix Ide platform explorting | Deploy the smart contract code    Deploy and run the transaction.  By selecting the environment - inject the MetaMask. |  |
| 4 | Open file explorer | Open the extracted file and click on the folder.    Open src, and search for utiles.    Open cmd enter commands  1.npm install   1. npm install bootstrap      1. npm start |  |
| 5 | {LOCALHOST  I  P ADDRESS | copy the address and open it to chrome so you can see the front end of your project. |  |

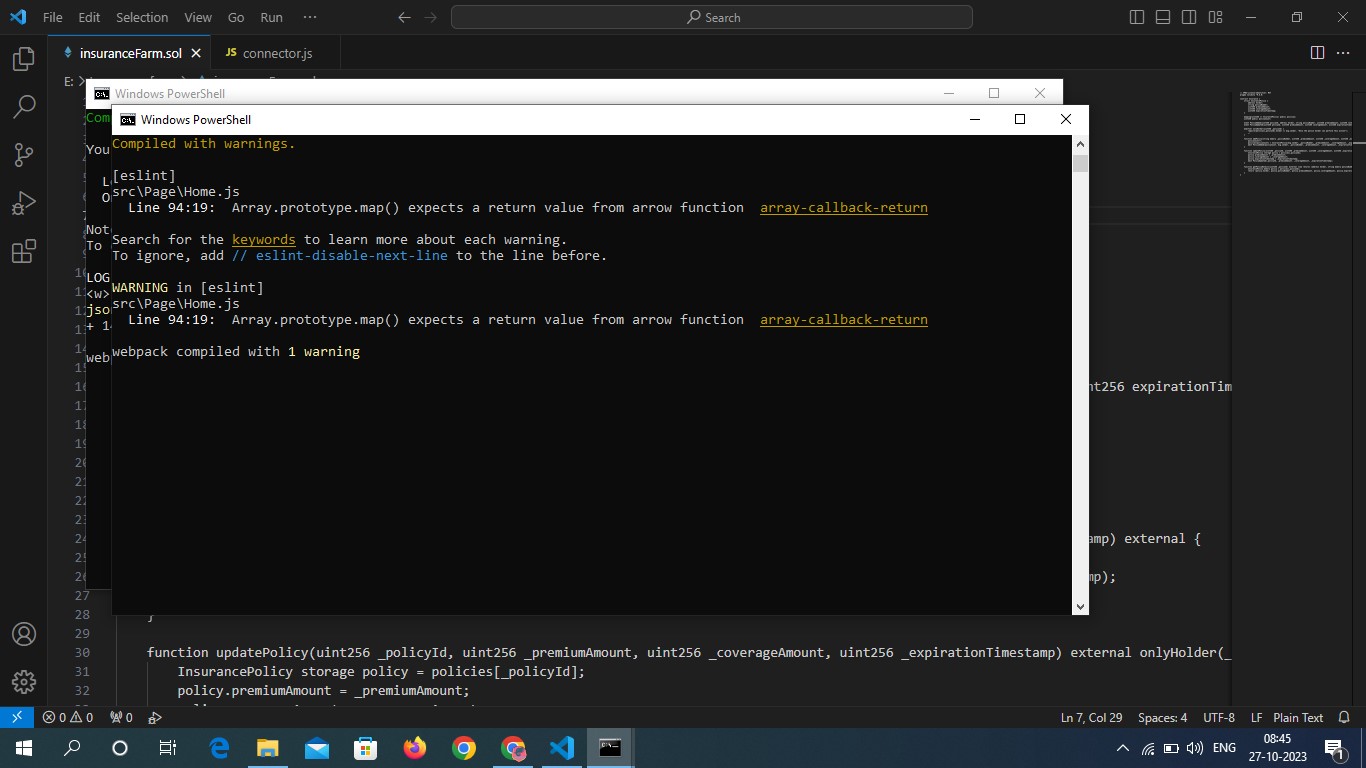
# 9. RESULTS

**9.1 OUTPUT SCREENSHOTS:**









1. **ADVANTAGE & DISADVANTAGE:**

**10.1 ADVANTAGE:**

**Increased transparency and trust:** Blockchain is a distributed ledger technology, which means that all transactions are recorded on a shared and secure network. This can help to increase transparency and trust between farmers and insurance companies, as all claims data is publicly accessible and immutable.

**Reduced costs and fraud:** Blockchain can help to reduce the costs and fraud associated with traditional insurance systems. For example, smart contracts can be used to automate the claims process, which can save time and money for both farmers and insurers. Additionally, blockchain can be used to track the provenance of crops and other agricultural products, which can help to reduce fraud.

**Improved access to insurance:** Blockchain can help to improve access to insurance for farmers, especially in developing countries. For example, blockchain-based microinsurance products can be offered to farmers at a lower cost and with less paperwork than traditional insurance products.

**Smart contracts:** Smart contracts can be used to automate the claims process, making it faster and easier for farmers to receive payouts. For example, a smart contract could be triggered to pay out a claim to a farmer if satellite data shows that their crops have been damaged by a drought.

**10.2 DISADVANTAGE:**

**Cost:** Implementing and maintaining a blockchain-based insurance system can be expensive. This is especially true for small farmers who may not have the financial resources to invest in new technology.

**Complexity:** Blockchain technology is complex and can be difficult to understand and use. This can be a barrier for farmers who may not have the technical expertise to use a blockchain-based insurance system.

**Lack of adoption:** Blockchain technology is still in its early stages of development and adoption. This means that there may not be a large number of blockchain-based insurance providers available, and farmers may have difficulty finding coverage that meets their needs.

**Regulatory uncertainty:** There is currently no clear regulatory framework for blockchain-based insurance. This uncertainty could make it difficult for farmers to trust blockchain-based insurance providers and could also delay the adoption of blockchain technology in the insurance industry.

**Data privacy concerns:** Blockchain technology is a distributed ledger system, which means that all data on the blockchain is publicly accessible. This could raise privacy concerns for farmers who are not comfortable with their personal and financial data being publicly accessible.

**Risk of fraud:** Blockchain technology is not immune to fraud. There have been cases of hackers stealing blockchain-based assets, and it is possible that hackers could also target blockchain-based insurance systems.

# 11. CONCLUSION

The "Farmer Insurance Using Blockchain" project represents an innovative and transformative approach to address the longstanding challenges of the agricultural insurance sector. By harnessing the power of blockchain technology, this project has the potential to revolutionize the way farmers access insurance, mitigate risks, and secure their livelihoods. In conclusion, let's summarize the key points of this project.The project's primary objective is to provide a reliable, transparent, and efficient insurance system tailored to the needs of farmers, offering them financial security and peace of mind in the face of agricultural risks.The system incorporates user registration, policy management, premium calculation, claims processing, smart contracts, payment integration, and a secure user interface. These features ensure a comprehensive and seamless insurance experience

|  |  |
| --- | --- |
|  |  |

# 12. FUTURE SCOPE

The future scope of the "Farmer Insurance Using Blockchain" project is promising, with several areas for potential expansion and improvement. Here are some future scope considerations.Explore the integration of Internet of Things (IoT) devices for real-time data collection. IoT sensors can provide valuable data on weather conditions, soil moisture, and crop health, allowing for more accurate risk assessment and claims processing implement artificial intelligence and machine learning algorithms to improve risk assessment, automate underwriting, and enhance claims processing. Integrate weather forecasting services to provide farmers with advanced notice of potential weather-related risks. This proactive approach can help farmers take preventive measures.

# 13. APPENDIX

**13.1 SOURCE CODE:**

|  |
| --- |
| // SPDX-License-Identifier: MIT pragma solidity ^0.8.0;  contract Insurance { struct InsurancePolicy { address holder; string policyNumber; uint256 premiumAmount; uint256 coverageAmount; uint256 expirationTimestamp;  } mapping(uint256 => InsurancePolicy) public policies; uint256 public policyCount;  event PolicyAdded(uint256 policyId, address holder, string policyNumber, uint256 premiumAmount, uint256 coverageAmount, uint256 expirationTimestamp); event PolicyUpdated(uint256 policyId, uint256 premiumAmount, uint256 coverageAmount, uint256 expirationTimestamp);  modifier onlyHolder(uint256 \_policyId) { require(policies[\_policyId].holder == msg.sender, "Only the policy holder can perform this action");  \_;  } function addPolicy(string memory \_policyNumber, uint256 \_premiumAmount, uint256 \_coverageAmount, uint256 \_expirationTimestamp) external { policyCount++; policies[policyCount] = InsurancePolicy(msg.sender, \_policyNumber,  \_premiumAmount, \_coverageAmount, \_expirationTimestamp);  emit PolicyAdded(policyCount, msg.sender, \_policyNumber,  \_premiumAmount, \_coverageAmount, \_expirationTimestamp);  } function updatePolicy(uint256 \_policyId, uint256 \_premiumAmount, uint256  \_coverageAmount, uint256 \_expirationTimestamp) external onlyHolder(\_policyId) {  InsurancePolicy storage policy = policies[\_policyId]; policy.premiumAmount = \_premiumAmount; policy.coverageAmount = \_coverageAmount; policy.expirationTimestamp = \_expirationTimestamp; emit PolicyUpdated(\_policyId, \_premiumAmount, \_coverageAmount, \_expirationTimestamp); |
| } function getPolicyDetails(uint256 \_policyId) external view returns (address holder, string memory policyNumber, uint256 premiumAmount, uint256 coverageAmount, uint256 expirationTimestamp) {  InsurancePolicy memory policy = policies[\_policyId]; return (policy.holder, policy.policyNumber, policy.premiumAmount, policy.coverageAmount, policy.expirationTimestamp);  }  } |

**13.2 GitHub & Project Demo Link:**