A PROJECT REPORT

On

ENHANCED AGRICULTURAL MANAGEMENT APPROACH FROM SUSTAINABLE CROP PRODUCTION TO SELLING INCLUDING CROP INSURANCE AND STUBBLE UTILIZATION FRAMEWORK USING BLOCK CHAIN SECURITY WITH IOT & MACHINE LEARNING ALGORITHMS

Submitted in partial fulfilment of the requirements

For the award of the degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

DIVVELA SINDHU VENKATA DURGA GOWRI (20JR1A0523)

CHATHARASUPALLI GAYATHRI (20JR1A0510)

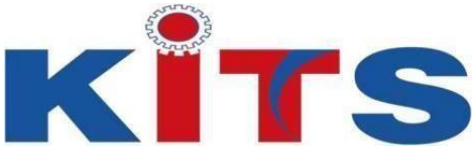
JAGARLAMUDI SAI SATHVIKA (20JR1A0528)

MATHE VIJAYA MERCY (21JR5A0504)

Under the guidance of

Dr. CHITTINENI ARUNA, M. Tech.,Ph.D

Professor & Head Administration, Dept. of CSE.



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES

(AUTONOMOUS)

(Approved by A.I.C.T.E New Delhi || Permanently Affiliated to JNTUK, Kakinada) || Accredited with 'A' Grade by NAAC || NBA Accreditation)

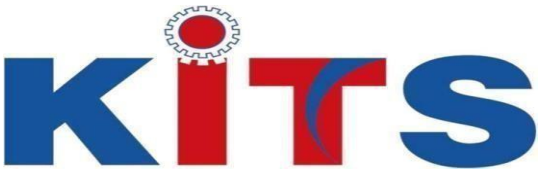
Vinjanampadu (V), Vatticherukuru (M), Guntur, A.P-522017.

April - 2024

KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES
(Autonomous)
(Approved by A.I.C.T.E New Delhi || Permanently Affiliated to JNTUK, Kakinada) ||
Accredited with 'A' Grade by NAAC || NBA Accreditation)

Vinjanampadu (Vil), Vatticherukuru (Md), Guntur (DT), A.P-522017.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



CERTIFICATE

This is to certify that this project work titled " **Enhanced Agriculture Management Approach From Sustainable Crop Production To Selling Including Crop Insurance And Stubble Utilization Framework Using Block Chain Security With IoT & Machine Learning Algorithms**" is the bonafide work of **Divvela Sindhu Venkata Durga Gowri (20JR1A0523)**, **Chatharasupalli Gayathri (20JR1A0510), Jagarlamudi Sai Sathvika (20JR1A0528), Mathe Vijaya Mercy (21JR5A0504),** who carried out the work under supervision of Dr. Chittineni Aruna, M.Tech.,Ph.D, Professor & Head Administration, Dept. of CSE, and submitted in the partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in Computer Science and Engineering through KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES(Autonomous) Affiliated to **JNTU-Kakinada** during the academic year 2023-2024.

HEAD OF THE DEPARTMENT **PROJECT GUIDE**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

We hereby inform that this main project entitled "**Enhanced Agriculture Management Approach From Sustainable Crop Production To Selling Including Crop Insurance And Stubble Utilization Framework Using Block Chain Security With IoT & Machine Learning Algorithms**" has been carried out and submitted in partial fulfilment for the award to the degree of **Bachelor of Technology** in Computer Science and Engineering to **JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA** under the guidance of **Dr. Chittineni Aruna,** Professor & Head-Administration, Department of Computer Science and Engineering. The work embodied in this project work is original and has not been submitted in part or full for any degree of this or any degree of any other university.

DIVVELA SINDHU VENKATA DURGA GOWRI	(20JR1A0523)
CHATHARASUPALLI GAYATHRI	(20JR1A0510)
JAGARLAMUDI SAI SATHVIKA	(20JR1A0528)
MATHE VIJAYA MERCY	(21JR5A0504)

ACKNOWLEDGEMENT

We would like to express our profound gratitude towards **Dr. Chittineni Aruna**, Professor & Head-Administration Department of COMPUTER SCIENCE AND ENGINEERING**,** who played a supervisory role to utmost perfection, enabled us to seek through our B. Tech main project and for guiding as an internal guide methodically and meticulously.

We express our gratitude towards all the faculty members and non-teaching faculty members, Department of COMPUTER SCIENCE AND ENGINEERING.

We are highly indebted to **Prof. R. RAMESH**, **Head of the Department,** Computer Science and Engineering for providing us all the necessary support.

We render our deep sense of gratitude to **Dr. P. BABU, Principal,** for permitting us to carry out our main project works.

We express our heartfelt thanks to **Dr.K.HARIBABU, principal,** for permitting us to complete our project work without any obstacles.

We would like to express our sincere thanks to Computer Science and Engineering staff for lending us their time to help us and complete the work successfully.

We are very much thankful to the **College Management** for their continuous support and facilities provided. We would also like to thank our staff, parents and friends for their enduring encouragement and assistance whenever required.

By

DIVVELA SINDHU VENKATA DURGA GOWRI (20JR1A0523)

CHATHARASUPALLI GAYATHRI (20JR1A0510)

JAGARLAMUDI SAI SATHVIKA (20JR1A0528)

MATHE VIJAYA MERCY (21JR5A0504)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To produce eminent and ethical Engineers and Managers for society by imparting quality professional education with emphasis on human values and holistic excellence.

INSTITUTION MISSION

- To incorporate benchmarked teaching and learning pedagogies in curriculum.
- To ensure all round development of students through judicious blend of curricular, co-curricular and extra-curricular activities.
- To support cross-cultural exchange of knowledge between industry and academy.
- To provide higher/continued education and researched opportunities to the employees of the institution.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a reputed centre in Computer Science & Engineering for quality, competency and social responsibility.

MISSION OF THE DEPARTMENT

- Strengthen the core competence with vibrant technological education in a congenial environment.
- Promote innovate research and development for the economic, social and environment.
- Inculcate professional behaviour, strong ethical values to meet the challenges in collaboration and lifelong learning.

Program Specific Outcomes (PSOs)

PSO1: Application Development

Able to develop the business solutions through Latest Software Techniques and tools for real time Applications.

PSO2: Professional and Leadership

Able to practice the profession with ethical leadership as an entrepreneur through participation in various events like Ideathon, Hackathon, project expos and workshops.

PSO3: Computing Paradigms

Ability to identify the evolutionary changes in computing using Data Sciences, Apps, Cloud computing and IoT.

Program Educational Objectives (PSOs)

Graduate of Computer Science and Engineering shall

PEO 1:

Domain Knowledge: Have a strong foundation in areas like mathematics, science and engineering fundamentals so as to enable them to solve and analyse engineering problems and prepare them to careers, R&D and studies of higher level.

PEO 2:

Professional Employment: Have an ability to analyse and understand the requirements of software, technical specifications required and provide novel engineering solutions to the problems associated with hardware and software.

PEO 3:

Higher Degrees: Have exposure to cutting edge technologies thereby making them to achieve excellence in the areas of their studies.

PEO 4:

Engineering Citizenship: Work in teams on multi-disciplinary projects with effective communication skills and leadership qualities.

PEO 5:

Lifelong Learning: Have a successful career wherein they strike a balance between ethical values and commercial values.

PROGRAM OUTCOMES (POS)

1. Engineering knowledge:

Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem analysis:

Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/development of solutions:

Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct investigations of complex problems:

Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern tool usage:

Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

6. The engineer and society:

Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and sustainability:

Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics:

Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work:

Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication:

Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance:

Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning:

Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Course Outcomes (COS)

CO421.1: Analyze the System of Examination and identify the problem.

CO421.2: Identify and classify the requirements.

CO421.3: Review the Related Literature.

CO421.4: Design and Modularize the project.

CO421.5: Construct, Integrate, Test and Implement the Project.

CO421.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes - Program Outcomes mapping

	1	2	3	4	5	6	7	8	9	10	11	12	PSO1	PSO2	PSO3
CO421.1	2	3		2		2	2						3		
CO421.2				3	2								3		2
CO421.3		3				2			3	3		1	2		1
CO421.4			2		3	2	2	2			1	2	3		2
CO421.5		2			2				2	2				2	
CO421.6	1					2		2	2	2	2			3	

3: High 2 : Medium 1: Low

Program Educational Objectives – Program Specific Outcomes correlation

	PSO1	PSO2	PSO3
PEO1	2	1	3
PEO2		3	2
PEO3	1	2	3
PEO4	3		2
PEO5	1	3	2

3: High 2: Medium 1: Low

CO-PO Mapping with Reasons:

CO421.1 is mapped with PO1, PO2 and PO4, PO6, PO7 as basic knowledge of Engineering and problem Analysis activities are highly essential to conduct examinations on existing systems which have been using in industries as a part of and to define the problem of proposed system.

CO421.2 is mapped with PO4 and PO5 as for identification, gathering analysis and classification of requirements for the proposed system, basic knowledge of engineering and Analysis steps along with complex problem analysis through the efforts of teamwork in order to meet the specific needs of the customer.

CO421.3 is mapped with PO2, PO6, PO9, PO10 and PO12 as to conduct the literature review and to examine the relevant systems to understand and identify the merits and demerits of each to enhance and develop the proposed as per the need.

CO421.4 is mapped with PO3, PO5, PO6, PO7, PO8, PO11 and PO12 because modularization and design of the project is needed after requirements elicitation. For modularization and design of the project, Basic knowledge of Engineering, Analysis capabilities, Design skills and communication is needed between team members as different modules are designed individually before integration.

CO421.5 is mapped with PO2, PO5, PO9 and PO10 as to construct the project latest technologies are needed. The development of project is done individually and in groups with well- defined communication by using the engineering and management principles.

CO421.6 is mapped with PO1, PO6, PO8, PO9, PO10 and PO11 because during and after completion of the project, documentation is needed along with proper methods of presentation through understanding and application of engineering and management principles, which in turn needs well defined communication between the team members with all the ethical values. Even the project development team defines the future enhancements as a part of the project development after identifying the scope of the project.

CO-PSOs Mapping with Reasons:

CO421.1 is mapped with **PSO1** as examining of existing systems and identification of the problem is a part of Application Development activity and identification of evolutionary changes in latest technologies and able to develop the applications using latest Technologies.

CO421.2 is mapped with **PSO1** and **PSO3** as identifying and classifying the requirements is a part of Application development and able to compute the evolutionary changes accordingly.

CO421.3 is mapped with **PSO1, PSO3** as review of literature is a part of application development activity by recognizing the computing technologies and able to compute the evolutionary changes accordingly.

CO421.4 is mapped with **PSO1, PSO3** because modularization and logical design is also a part of Application development and able to compute the evolutionary changes accordingly in Blockchain Technology.

CO421.5 is mapped with **PSO1** as Testing, Development and Integration of project activities are part of Application development and follows ethical principles.

CO421.6 is mapped with **PSO2** as for project documentation and presentation; the project team members apply the professional and leadership quality.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Agriculture, pivotal for a country with the largest global population, faces challenges in maintaining an efficient supply chain and ensuring data transparency. This initiative aims to revolutionize the agricultural sector by leveraging blockchain technology. The implementation of blockchain addresses issues such as fraud prevention, enhancement of food safety, quality management, and ensuring equitable prices for farmers. Incorporating IoT sensors generates valuable data reflecting crop quality, while machine learning algorithms analyze this data for effective crop disease detection. The introduction of a bidding smart contract streamlines the crop-selling process, facilitating insurance claims for farmers, with all pertinent information securely stored in an immutable and transparent ledger. Additionally, optimizing the treatment of crop remnants contributes to increased farmer income. By seamlessly integrating these technologies with agriculture, overall efficiency and the potential of the agricultural sector are significantly heightened.

CHAPTER-1: INTRODUCTION

1.1 Introduction of the project

The agricultural sector, integral to sustaining the burgeoning global population, confronts formidable challenges in streamlining its supply chain and ensuring transparent data practices. In response to these pressing issues, this initiative endeavours to usher in a new era for agriculture by harnessing the transformative power of blockchain technology. Blockchain implementation serves as a cornerstone in overcoming fraud related challenges, food safety, quality management, and fair pricing for farmers.

As part of this initiative, the integration of Internet of Things (IoT) sensors plays a pivotal role in revolutionizing real-time data access for farmers. Key parameters such as soil moisture levels, temperature, air quality, soil fertility, and light intensity contribute significantly to the determination of crop quality. This data-driven approach, enhanced by machine learning algorithms for precise crop disease detection, not only ensures optimal agricultural practices but also facilitates the implementation of a bidding smart contract, simplifying the crop-selling process. The proposed system encompasses a comprehensive data set, covering crucial soil parameters, moisture levels, and atmospheric conditions.

Farmers receive this data at regular intervals, enabling informed decision-making for irrigation, disease prevention, and crop-selling strategies. Through blockchain-compatible storage, data is not only secure but also transparent and traceable, enhancing the overall integrity of the agricultural ecosystem. Moreover, the initiative introduces innovative methods for crop selling, including individual buyer transactions, bidding smart contracts, and a digital market interface. The strategic repurposing of crop residues, or stubble, further contributes to farmer income by selling them to bioenergy-producing industries.

The blockchain records transparent and verifiable transaction histories for stubble sales, ensuring accountability. Regarding risk mitigation and financial security, the initiative provides farmers with a catalogue of insurance schemes tailored to individual needs, addressing potential risks associated with natural calamities. All critical data, spanning crop parameters, disease identification results, crop selling transactions, stubble sales, and insurance schemes, is securely stored in the blockchain. This comprehensive approach ensures immutability, transparency, and traceability, ultimately enhancing the integrity and reliability of the agricultural system. Through the seamless integration of

these technologies, the initiative aims to significantly heighten overall efficiency and unlock the untapped potential of the agricultural sector.

1.2 Existing System

Below are some of the existing systems that we gone through:

In this study, precision agriculture is explored through the integration of IoT, Machine Learning, and blockchain technologies on a permission-based blockchain network. The implementation, utilizing IBM dependencies, ensures secure and efficient communication among interconnected elements in agriculture. The research successfully demonstrates the incorporation of IoT data for crop recommendations, secured by blockchain, and envisions future enhancements, such as cloud deployment for broader accessibility and the exploration of public/private sub-networks for specific agricultural groups within the main network. The study highlights the potential of technology convergence in optimizing resource utilization and communication in precision agriculture.

By leveraging IoT sensors for data collection and machine learning for crop analysis, blockchain integration streamlines the supply chain, reducing middlemen, transaction costs, and enhancing farmer profits. The proposed methodology ensures secure, transparent, and efficient transactions, facilitating easy insurance claims for farmers. In conclusion, blockchain has the potential to revolutionize the entire food and agriculture supply chain, fostering trust, reducing fraud, and increasing overall efficiency. Future advancements are expected to introduce even more cutting-edge applications in the agriculture sector.

The proposed web application serves as a bridge between farmers and consumers, offering farmers an additional income source by enabling direct product sales through electronic media. By empowering farmers to manage complete payments independently, it eliminates middlemen involvement prevalent in traditional mandi systems. The platform caters to two primary users: farmers, who showcase crop information and rates on the website, and consumers, who place orders with a 40% advance payment.

The application's workflow involves HTML, CSS3, and Vanilla JavaScript for frontend design and functionality, integrated with Django for backend support. The system undergoes testing and improvements before delivering results, ensuring a seamless and efficient agricultural marketplace.

1.3 Problems identified in Existing System

Here are some potential problems in the existing system of your agricultural project:

Limited Data Accessibility:

- Farmers may face challenges in accessing real-time data on soil parameters and atmospheric conditions due to limited connectivity or outdated technology.

Manual Irrigation Control:

- The reliance on manual intervention for irrigation practices can lead to inefficiencies and inconsistencies in water usage, potentially resulting in over or under watering.

Disease Detection Limitations:

- The current disease detection methods may be inadequate, resulting in delayed or inaccurate diagnosis of crop diseases, leading to potential yield losses.

Inefficient Selling Mechanisms:

- The existing selling mechanisms may lack transparency and efficiency, causing delays and disputes in transactions between farmers and buyers.

Underutilization of Stubble:

- Crop residues (stubble) may be underutilized, leading to missed opportunities for additional income generation and sustainable agricultural practices.

Limited Insurance Coverage:

- Existing insurance schemes may not adequately cover farmers' needs, leaving them vulnerable to financial losses due to natural calamities or crop failures.

Complexity in User Interface:

- The user interface may be complex and difficult to navigate, leading to user frustration and reduced adoption of the system.

Addressing these problems can lead to significant improvements in the effectiveness and efficiency of your agricultural management system, ultimately benefiting farmers and enhancing agricultural practices.

1.4 Proposed System

The proposed agricultural system encompasses a comprehensive dataset compiled systematically, covering soil parameters such as nitrogen, phosphorus, and potassium levels, along with moisture and atmospheric parameters like air quality, temperature, and light intensity. This data is presented to farmers at 2-hour intervals and temporarily stored in a centralized repository for subsequent averaging on daily, monthly, and blockchain-

compatible block levels. Leveraging moisture data, an automated irrigation system is implemented to optimize water resource utilization by automatically activating or deactivating the water motor without direct farmer intervention. Furthermore, a sophisticated disease identification system employs machine learning algorithms to detect crop diseases, providing farmers with precise organic remedies. The results from disease identification are securely stored in the blockchain network, ensuring transparency and traceability.

For crop selling, three distinct methods are proposed: individual buyer transactions, bidding smart contracts, and a digital market interface. Additionally, crop residues, or stubble, are repurposed by selling them to bioenergy-producing industries nearby, contributing to additional income for farmers. The conclusive report on stubble sales is recorded in the blockchain, ensuring a transparent and verifiable transaction history. Moreover, a catalog of insurance schemes tailored to individual needs is made available to farmers, promoting risk mitigation and financial security. Farmers can apply for relevant insurance schemes based on their specific requirements. All critical data, including crop parameters, disease identification results, crop selling transactions, stubble sales, and insurance schemes, is securely stored in the blockchain, ensuring immutability, transparency, and traceability, thereby enhancing the overall integrity and reliability of the agricultural system.

1.5 Benefits of our System

- Increased Efficiency: By leveraging IoT, blockchain, and machine learning technologies, the project streamlines agricultural processes, leading to higher productivity and resource optimization.
- Enhanced Crop Quality: Through real-time monitoring and disease identification, farmers can promptly address issues, resulting in improved crop health and yield.
- Financial Security: The implementation of tailored insurance schemes provides farmers with financial protection against crop losses due to natural calamities, ensuring stability in their livelihoods.
- Sustainable Practices: The project promotes organic remedies, stubble monetization, and transparent selling mechanisms, contributing to environmentally friendly and sustainable farming practices.
- Market Access and Transparency: With diversified crop selling mechanisms and a user-friendly digital market interface, farmers gain access to a broader market and benefit

from transparent transactions, ensuring fair pricing and improved market visibility.

- Data-Driven Decision Making: Comprehensive monthly averaging and blockchain-logged transaction history provide valuable insights to farmers, empowering them to make informed decisions and optimize their farming strategies.

1.6 Potential Users

In this project, the potential users are

- Farmers
- Crop Buyers/Stakeholders
- Bioenergy-producing Industries
- Insurance Providers
- Government Agencies (Agricultural and Environmental)
- End Consumers (for direct digital market transactions)
- Environmental Agencies (related to stubble management)

1.7 Unique features of the System

- Organic Remediation for disease identified
- Diversified Crop Selling Mechanisms
- Stubble Monetization Strategy
- Blockchain-Logged Transaction History
- Tailored Insurance Scheme Catalog
- User-Friendly Digital Market Interface
- Comprehensive Monthly Averaging

1.8 Demand of Project

Here are the points outlining the demand for your agricultural management system:

Efficiency Enhancement:

- Farmers seek solutions to streamline operations and improve efficiency.
- Your system offers tools to automate tasks such as data collection and irrigation, meeting the demand for efficiency enhancements.

Increased Productivity:

- Farmers aim to maximize yields and improve crop quality.
- Your system provides real-time data and analytics to aid decision-making, fulfilling the demand for increased productivity.

Sustainable Practices:

- There is a growing demand for sustainable agricultural practices.
- Your system promotes sustainability through features like stubble utilization and organic disease management, addressing this demand.

Transparency and Trust:

- Farmers and buyers seek transparent and trustworthy platforms for transactions.
- Your system leverages blockchain technology to ensure transparency and security, meeting the demand for trusted agricultural platforms.

Risk Mitigation:

- Farmers face risks such as crop diseases and market fluctuations.
- Your system's insurance scheme management feature provides risk mitigation, fulfilling the demand for financial protection.

Accessibility and Ease of Use:

- There is a demand for user-friendly agricultural management systems.
- Your system offers a simple interface and accessible features, making it suitable for users of all skill levels.

Addressing these points ensures that your agricultural management system meets the diverse demands of farmers and stakeholders in the agricultural industry, ultimately driving its adoption and success.

1.9 Protection of Idea

Patent: Obtaining a patent is a critical step to safeguard your unique solution and prevent others from commercially exploiting it without your permission. In addition to the patent application number **202441029689** for our unique solution, we have successfully obtained a patent with the title **"Enhanced Agriculture Management Approach From Sustainable Crop Production To Selling Including Crop Insurance And Stubble Utilization Framework Using Block Chain Security With IoT & Machine Learning Algorithms"** This patent provides legal protection for our invention, ensuring that others cannot commercially exploit it without our permission.

CHAPTER – 2: ANALYSIS

2.1 Literature Review

The integration of IoT, Machine Learning, and Blockchain in precision agriculture, as explored by Akhila Susan Babu and Supriya M in 2022, promises to revolutionize farming practices by optimizing resource usage, enhancing product quality, and ensuring transparent transactions throughout the supply chain. Their comprehensive review underscores the potential of these technologies to empower farmers with data-driven insights and secure, traceable transactions, paving the way for a more sustainable and efficient agricultural ecosystem.

The work by Hajar Moudoud, Soumaya Cherkaoui, and Lyes Khoukhi introduces LC4IoT in 2019, a specialized blockchain architecture tailored for supply chains integrating distributed IoT entities. Their innovative approach incorporates a lightweight consensus mechanism, effectively reducing computational requirements, storage overhead, and latency, thereby enhancing the efficiency and scalability of supply chain operations.

M. Harini, D. Dhinakaran, D. Prabhu, S. M. Udhaya Sankar, V. Pooja, and P. Kokila Sruthi in 2023, explore blockchain's impact on India's agriculture. Their study emphasizes enhanced supply chain management, transparency, and efficiency through IoT data, machine learning, and smart contracts, offering insights for sustainable growth.

S. Madumidha, P. Siva Ranjani, U. Vandhana, and B. Venmuhilan in 2019, propose a decentralized blockchain traceability system for agriculture food supply chains. Their focus on transparency, security, and traceability aims to enhance community engagement and satisfaction, ensuring trust and accountability from providers to consumers in the agricultural ecosystem.

Miguel Pincheira Caro, Muhammad Salek Ali, Massimo Vecchio, and Raffaele Giaffreda in 2018,present AgriBlockIoT, a decentralized blockchain-based traceability solution for **Agri-Food supply chains**. Seamlessly integrating IoT devices, it tackles data integrity and transparency issues. The study assesses Ethereum and Hyperledger Sawtooth implementations' performance in a comprehensive from-farm-to-fork use-case evaluation.

Ms. S. Madumidha, Dr. P. Siva Ranjani, Ms. S. Sree Varsinee, and Ms. P.S. Sundari in 2019, explore the integration of blockchain and IoT to improve supply chain transparency and efficiency. They address traditional logistics limitations and examine the synergies, advantages, and challenges of this combined approach, offering insights into enhancing modern agricultural practices.

Ilhaam A. Omar, Raja Jayaraman, Khaled Salah, and Jiju Antony in 2023, present an innovative blockchain-based crop index insurance solution. Their system ensures transparency, reduces fraud, expedites claim processing, and enhances affordability, promising to revolutionize crop insurance for farmers worldwide.

H. Binici, **M. Eken, M. Kara, and M. Dolaz in 2013**, investigate the use of sunflower stalks, cotton textile **waste,** and **stubble** as composite materials for insulation production. Their study aims to decrease heating and cooling expenses while tackling environmental concerns related to sunflower stem disposal.

Kavita Saini, **Ishika Mishra, and Shreya Srivastava in 2021**, examine the significance of e-commerce in enhancing business profitability. Their paper delves into the creation of an **e-commerce platform** for Indian farmers, aiming to eliminate intermediaries and overcome challenges through careful planning, domain modeling, and architectural patterns.

Aditya Ghodke, Ajay Kokare, Rakesh Shinde, Akshay Marathe, and Prof. S. S. Kashid in 2022,introduce an **E-Auction system** empowering farmers with direct communication and independent pricing for crop products. By eliminating intermediaries, it ensures fair market access through a bid-based mechanism, supplemented by GPS navigation for successful bidders, facilitating efficient and transparent transactions.

Aloysius Chiong Zhen Quan, Khairunnisa binti Sufian, Ng Sing Woei, Lai Li Ying, and Amanda Ling Tzi Yun in 2021, introduce FarmBid, an integrated e-commerce platform addressing contract farming challenges in Malaysia. By facilitating direct sales and offering standardized delivery services, it empowers farmers, enhances their bargaining power, and ultimately improves their income prospects.

Karthiga M, Srivarshan M, Danushmathi P, and Sharmila S in 2023,introduce a web-based program empowering farmers to establish fair crop prices via an online auction. Facilitated by a machine learning system ensuring market value accuracy, the platform, exclusive to registered farmers, fosters transparent and competitive bidding, ensuring optimal returns and offering decision support and expert assistance through a user-friendly interface.

Rahul Talreja, Rohan Chouksey, and Sushma Verma in 2020, present a farmer's portal leveraging blockchain technology and Python programming. This integration ensures secure and immutable recording of crop transactions, enhancing transparency and economic efficiency in agricultural operations.

Pratap S. Birthal, Awadhesh K. Jha, and Harvinder Singh in 2014, examine strategies like cooperatives, growers' associations, and contract farming to improve smallholders' market access for high-value food commodities. They stress the necessity for policy support to fortify these linkages, thus promoting inclusive growth in the agricultural sector. **Mrs. Manisha Bhende, Ms. Mohini S. Avatade, Mrs. Suvarna Patil, Mrs. Pooja Mishra, Ms. Pooja Prasad, and Mr. Shubham Shewalkar in 2018,**propose a digital market platform aiming to integrate farmers, merchants, government, and end users to address challenges faced by Indian farmers in securing fair profits. Leveraging mobile-based Android and web-based Java applications, along with KNN and Haversine algorithms, the system enables transparent transactions across market layers, fostering informed decision-making and government oversight. However, dynamic transportation record tracking poses a notable challenge.

2.1.1. Review Findings

Here are some review findings for the existing agricultural management system:

- Reviewers noted that accessing real-time data on soil parameters and atmospheric conditions was challenging due to connectivity issues and outdated technology.
- Farmers expressed frustration with the limited availability of data, hindering their ability to make informed decisions about crop management.
- Users highlighted the reliance on manual intervention for irrigation practices as a major drawback.
- Manual irrigation control led to inefficiencies and inconsistencies in water usage, resulting in suboptimal crop growth and yield.
- Reviewers found that the existing disease detection methods were not always accurate or timely.
- Delayed or inaccurate diagnosis of crop diseases resulted in increased risk of crop loss and reduced productivity.
- Users reported inefficiencies in the existing selling mechanisms, such as delays and disputes in transactions.
- Lack of transparency and reliability in the selling process posed challenges for both farmers and buyers.

- It was observed that crop residues (stubble) were often underutilized, representing a missed opportunity for additional income generation.
- Reviewers noted a lack of effective mechanisms for stubble utilization, leading to environmental and economic inefficiencies.
- Users expressed concerns about the limited coverage and flexibility of existing insurance schemes.
- Some farmers felt that the available insurance options did not adequately protect against various risks, leaving them financially vulnerable.
- Reviewers found the user interface to be complex and difficult to navigate, especially for users with limited technical expertise.
- Complicated user interface design led to user frustration and reduced system adoption rates.

These review findings highlight the shortcomings and areas for improvement in the existing agricultural management system. Addressing these issues can lead to a more efficient, reliable, and user-friendly system that better serves the needs of farmers and stakeholders in the agriculture industry.

2.1.2. Objectives of the system

- Leveraging IoT sensors for real-time monitoring of soil parameters, atmospheric conditions, and crop health indicators.
- Implementing blockchain technology to secure and transparently record transactions, ensuring trust and integrity in agricultural trade.
- Integrating machine learning algorithms for disease identification and recommendation of organic remedies, enhancing crop management practices.
- Offering diversified crop selling mechanisms including individual transactions, smart contracts, and digital market interfaces to cater to varied farmer and buyer needs.
- Providing tailored insurance schemes to mitigate financial risks associated with crop loss due to natural calamities, promoting stability within the farming community.
- Facilitating stubble utilization strategies to convert crop residues into valuable resources such as bioenergy or organic fertilizer, minimizing waste and promoting sustainability.
- Empowering farmers with user-friendly interfaces, real-time insights, and decision

support tools for optimized agricultural practices and increased profitability.

- Collaborating with stakeholders including government agencies, insurance providers, and bioenergy-producing industries to ensure comprehensive support and integration of resources.
- Promoting environmental conservation through water-efficient irrigation practices, reduced chemical usage, and responsible stubble management.
- Enhancing market access and transparency through digital platforms, providing farmers with greater control over their produce and fair market pricing.

2.2 Requirements analysis

Requirement Analysis, also known as Requirement Engineering, is the process of defining user expectations for new software being built or modified. Requirements analysis encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product or project, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements.

2.2.1. Functional requirements

It is a useful document which describes functions, appearance, purpose and requested outputs of the software. It allows you to structure all the information regarding an application, let's break down the software and hardware requirements for each stage:

Functional requirements analysis for your agricultural management system involves identifying and documenting the specific features and functionalities that the system must provide to meet the needs of users and stakeholders. Here's an outline of functional requirements for your project:

Data Collection and Monitoring:

- The system shall collect real-time data on soil parameters (fertility, moisture) and atmospheric conditions (air quality, temperature, light intensity) using IoT sensors.
- It shall display the collected data to farmers through a user-friendly interface for monitoring and analysis.

Automation Irrigation Control:

- The system shall use moisture data to automate the control of water motors for irrigation practices.
- It shall optimize irrigation schedules and water usage based on real-time soil moisture levels.

Disease Identification and Remediation:

- The system shall implement machine learning algorithms to detect crop diseases based on collected data.
- It shall provide farmers with organic remedy recommendations for disease management, including treatment options and dosage.

Crop Selling Mechanisms:

- The system shall offer multiple methods for selling crops, including individual buyer transactions, smart contract-based bidding, and a digital market interface.
- It shall facilitate transparent and efficient crop selling processes, including price negotiation and transaction settlement.

Stubble Utilization:

- The system shall enable the selling of crop residues (stubble) to bioenergy-producing industries.
- It shall ensure transparent stubble sales transactions and securely store related data in the blockchain for traceability.

Insurance Scheme Management:

- The system shall list various insurance schemes tailored to farmers' needs, providing financial protection against natural calamities.
- It shall enable farmers to securely store final insurance scheme reports in the blockchain for transparency and data integrity.

User Management and Authentication:

- The system shall allow users to register and create accounts with unique credentials (ID, password).
- It shall authenticate users' credentials during the login process to ensure secure access to the system.

Profile Management:

- The system shall enable users to manage their profiles, including personal information, preferences, and notification settings.

- It shall allow users to update and modify their profiles as needed.

Reporting and Analytics:

- The system shall generate reports and analytics based on collected data, including crop health status, yield predictions, and financial summaries.
- It shall provide visualizations and insights to help users make informed decisions and optimize agricultural practices.

Integration and Compatibility:

- The system shall integrate seamlessly with existing agricultural infrastructure and technologies, including IoT sensors and blockchain platforms.
- It shall be compatible with various devices and platforms, ensuring accessibility and usability for users.

By defining these functional requirements, can ensure that our agricultural management system effectively addresses the needs of users and stakeholders, providing valuable features and functionalities to optimize agricultural practices and improve productivity.

2.2.2. User requirements

User Requirements are used as the primary input for creating system requirements. These requirements are used for objective and testing of product information.

- Real-time monitoring of soil parameters and atmospheric conditions for informed decision-making.
- Seamless integration of organic disease remedies for effective crop management.
- Transparent and flexible crop selling mechanisms catering to diverse farmer and buyer needs.
- Access to tailored insurance schemes ensuring financial security against crop loss.
- Intuitive digital interfaces facilitating easy interaction and transaction processing for all stakeholders.

2.2.3. Non-functional requirements

Apart from functional requirements, we also have some non-functional requirements. **Non-functional requirements or NFRs** are a set of specifications that describe the system's operation capabilities and constraints and attempt to improve its functionality. These are basically the requirements that outline how well it will operate including things like speed,

security, reliability, data integrity, etc. **Non-Functional Requirement** (NFRs) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system.

- **Security:** Implement robust data encryption techniques to protect sensitive information such as farmer data, transaction details, and crop monitoring data from unauthorized access or tampering.
- **Reliability:** Ensure the system operates consistently and accurately under varying conditions, minimizing downtime and errors in data collection, analysis, and transaction processing.
- **Portability:** Design the system to be compatible with various devices and platforms, allowing farmers and stakeholders to access and interact with the system seamlessly regardless of their preferred hardware or operating system.
- **Maintainability:** Develop the system with modular and well-documented code, facilitating easier maintenance, updates, and enhancements as technology evolves or new requirements arise.
- **Privacy:** Implement privacy-preserving measures to safeguard personal and confidential information, adhering to data protection regulations and ensuring farmers' privacy rights are respected throughout the system's lifecycle.
- **Efficiency:** Optimize system performance to minimize resource usage, response times, and energy consumption, ensuring smooth operation even in resource-constrained environments and maximizing the system's overall efficiency.

2.2.4. System requirements

System requirements are the configuration that a system must have in order for a hardware or software application to run smoothly and efficiently. Failure to meet these requirements can result in installation problems or performance problems. They are often provided to consumers in complete detail. System requirements often indicate the minimum and the recommended configuration. The former is the most basic requirement, enough for a product to install or run, but performance is not guaranteed to be optimal. System requirements are also known as minimum system requirements. They are broadly classified into 2 categories.

Software requirements for a system are the description of what the system should do, the service or services that it provides and the constraints on its operation. The IEEE Standard Glossary of Software Engineering Terminology defines a requirement as:

- A condition or capability needed by a user to solve a problem or achieve an objective.
- A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.

A documented representation of a condition or capability as 1 or 2 Software Requirements of our system are:

- Programming Language and Framework
- Blockchain Implementation
- Database
- Web Development Tools
- RESTful API
- Dependencies and Packages

Hardware system requirements often specify the operating system version, processor type, memory size, available disk space and additional peripherals, if any, needed. The hardware requirements of our system are as follows.

- Windows/Linux/Mac etc., any OS can be used and our system is independent of theirversions.
- I3 processor version is enough
- Memory size of 1Gb disk is enough
- Ram can be of 2Gb
- Sensors like Temperature Sensors, Humidity Sensors, Moisture Sensor
- CO2 Sensor, O2 Sensor
- Arduino board
- Connecting Wires

2.3. Module description

It describes the major functionalities of each module.

Register and Login Module:

- Allows users, including farmers and stakeholders, to create accounts securely.

- Manages user authentication and authorization processes for accessing the system's functionalities.
- Ensures data privacy and security through encrypted login credentials and authentication protocols.

Data Collection Module:

- Utilizes IoT sensors to collect real-time data on soil parameters and atmospheric conditions.
- Transmits collected data to the central system for analysis and visualization.

Crop Monitoring and Disease Identification Module:

- Analyzes collected data using machine learning algorithms to monitor crop health and detect diseases.
- Provides farmers with timely alerts and recommendations for organic remedies.

Irrigation Control Module:

- Utilizes moisture data to automate irrigation systems, optimizing water usage and crop health.
- Allows farmers to remotely control and monitor irrigation processes through a user-friendly interface.

Crop Selling Module:

- Offers various selling mechanisms including individual transactions, smart contracts, and a digital market interface.
- Facilitates transparent and efficient transactions between farmers and buyers.

Stubble Utilization Module:

- Manages stubble sales transactions with bioenergy-producing industries, providing farmers with an additional source of income.
- Ensures transparency and securely logs stubble-related data on the blockchain.

Insurance Scheme Management Module:

- Lists and manages tailored insurance schemes for farmers' financial protection against crop loss.
- Stores insurance-related data securely on the blockchain for transparency and integrity.

2.4. Feasibility Study

A feasibility study is an evaluation of a proposal designed to determine the difficulty in carrying out a designated task. Generally, a feasibility study precedes technical

development and project implementation. In other words, a feasibility study is an evaluation or analysis of the potential impact of a proposed project. Preliminary investigation examines project feasibility; the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test Technical, Operational and Behavioral feasibility for adding new modules and debugging old running systems.

- Technical feasibility
- Operational feasibility
- Economical feasibility

2.4.1. Technical feasibility

This study is carried out to check the technical feasibility i.e., the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. 'Evaluating technical feasibility is the trickiest part of a feasibility study. A number of issues have to be considered while doing a technical analysis. The usage of this application over a cloud environment simply needs an uninterrupted and uninterrupted internet connection.

The technical requirements of this project like Operating System support that are related to the end users are affordable by every user. The developer also need not have any advanced knowledge of all our technical aspects. So based on these circumstances we can say that the project supports technical feasibility

2.4.2. Operational feasibility

It is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of a system development. The proposed system solves all the problems in the existing system on the basis of time, cost, risk, etc.… In this project all the data is managed by the admin .As the manager will already have enough knowledge, he no longer needs to be trained to utilize the software. In this way this project supports Operational feasibility also.

2.4.3. Economical feasibility

Assessing the economical feasibility of your agricultural management system involves a detailed examination of both costs and potential benefits. Initial costs may include hardware procurement, software development, infrastructure setup, and personnel training. For instance, the acquisition of IoT sensors and related equipment could amount to approximately 10,000 to 20,000, while software development costs may range from 50,000

to 10,000 depending on the complexity of the system. Infrastructure setup, including server deployment and database configuration, might require an additional investment of 20,000 to 30,000. Personnel training costs can vary but may average around 5,000 to 10,000 to ensure users are proficient in utilizing the system effectively.

On the benefits side, potential savings and revenue generation opportunities should be considered. For example, the system's optimization of irrigation practices could lead to significant water savings, potentially amounting to thousands of dollars annually depending on farm size and water usage. Similarly, improved disease management and crop monitoring could result in higher yields and reduced crop losses, translating into increased revenue for farmers. Additionally, transparent and efficient selling mechanisms facilitated by the system could lead to enhanced market access and better pricing for agricultural products.

2.5 Process Model Used

The process model which would be helpful for the development of the proposed system is the agile process model. The Agile model was primarily designed to help a project to adapt to change requests quickly. So, the main aim of the Agile model is to facilitate quick project completion. To accomplish this task agility is required. Agility is achieved by fitting the process to the project, removing activities that may not be essential for a specific project. Also, anything that is a waste of time and effort is avoided. Actually, Agile model refers to a group of development processes. These processes share some basic characteristics but do have certain subtle differences among themselves.

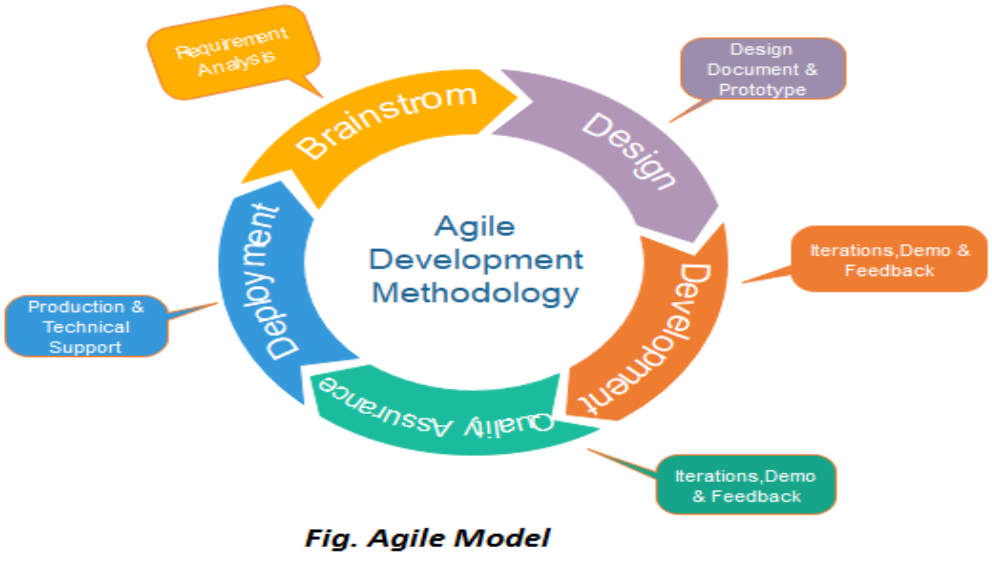


Fig-2.1: Process of Agile Model

Here's how the Agile process model can be applied to your project:

Project Initiation:

- Define project goals, objectives, and requirements based on stakeholder input and feasibility studies.
- Establish the core team members, including developers, domain experts, and stakeholders.

Initial Planning:

- Conduct a high-level estimation of project scope, timelines, and resource requirements.
- Prioritize project features and functionalities based on their importance and impact on the end users.

Iteration Planning:

- Break down the project scope into smaller, manageable chunks or iterations, typically lasting 1-4 weeks.
- Collaboratively plan each iteration with the development team, identifying specific tasks and goals to achieve.

Development Iterations:

- Implement the planned features and functionalities during each iteration.
- Conduct regular meetings, such as daily stand-ups, to track progress, discuss challenges, and adjust plans as needed.
- Continuously integrate and test the developed features to ensure quality and functionality.

Review and Feedback:

- At the end of each iteration, review the implemented features with stakeholders and gather feedback.
- Incorporate feedback into subsequent iterations, iterating on features based on user input and evolving requirements.

Continuous Integration and Deployment:

- Integrate newly developed features into the main codebase on a regular basis to maintain a stable and up-to-date system.
- Automate testing and deployment processes to streamline development and ensure consistent quality.

Retrospective and Improvement:

- Conduct regular retrospectives at the end of each iteration to reflect on the team's performance and identify areas for improvement.
- Implement process improvements and adjustments based on retrospective findings to enhance team productivity and efficiency.

Completion and Release:

- Continue iterating on features and functionalities until the project goals are met.
- Plan for a final release of the system, ensuring all requirements are fulfilled and the product is ready for deployment.

By adopting an Agile software development process model, our team can effectively manage the complexity of the project, adapt to changing requirements, and deliver a high-quality agricultural management system that meets the needs of users and stakeholders.

2.6 Hardware and Software Requirements

Software Requirements:

- Programming Language and Frameworks
- Blockchain Implementation
- Database
- Web Development Tools
- RESTful API
- Dependencies and Packages

Hardware Requirements:

- Processor
- RAM - 4 GB (min)
- Hard Disk - 20 GB

2.7. SRS Specification

- An SRS is basically an organization and understanding of a client and system requirements and dependencies at a particular point in time.
- The SRS document itself states in precise and explicit language those functions and capabilities a software system must provide, as well as states any required constraints by which the system must abide
- Functional requirements and non-functional requirements are two essential categories of requirements included in a Software Requirements Specification (SRS) document.

2.7.1 Functional Requirements

It is a useful document which describes functions, appearance, purpose and requested outputs of the software. It allows you to structure all the information regarding an application, let's break down the software and hardware requirements for each stage:

Functional requirements analysis for your agricultural management system involves identifying and documenting the specific features and functionalities that the system must provide to meet the needs of users and stakeholders. Here's an outline of functional requirements for your project:

Data Collection and Monitoring:

Fig-2.2: Data Collection

- The system shall collect real-time data on soil parameters (fertility, moisture) and atmospheric conditions (air quality, temperature, light intensity) using IoT sensors.
- It shall display the collected data to farmers through a user-friendly interface for monitoring and analysis.

Automation Irrigation Control:

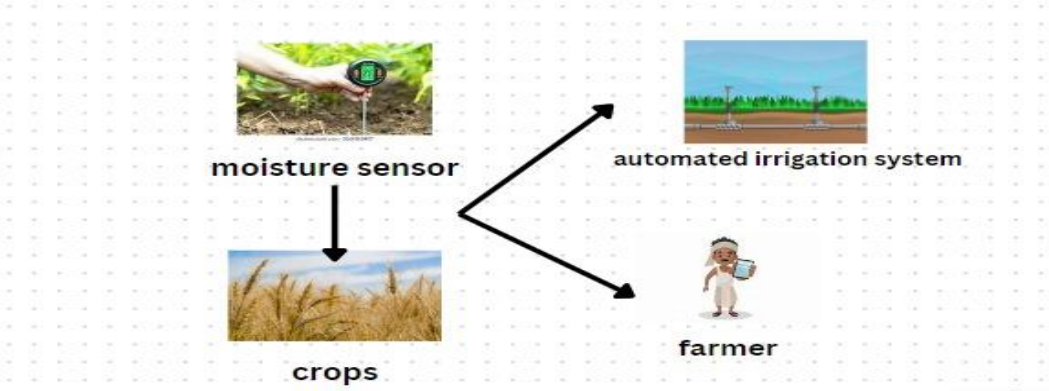


Fig-2.3: Irrigation Control

- The system shall use moisture data to automate the control of water motors for irrigation practices.

- It shall optimize irrigation schedules and water usage based on real-time soil moisture levels.

Disease Identification and Remediation:

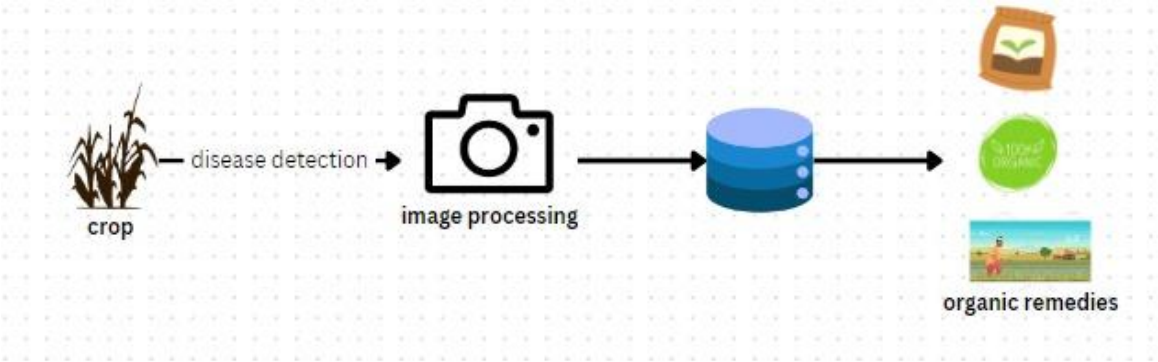


Fig-2.4: Disease Identification

- The system shall implement machine learning algorithms to detect crop diseases based on collected data.
- It shall provide farmers with organic remedy recommendations for disease management, including treatment options and dosage.

Crop Selling Mechanisms:

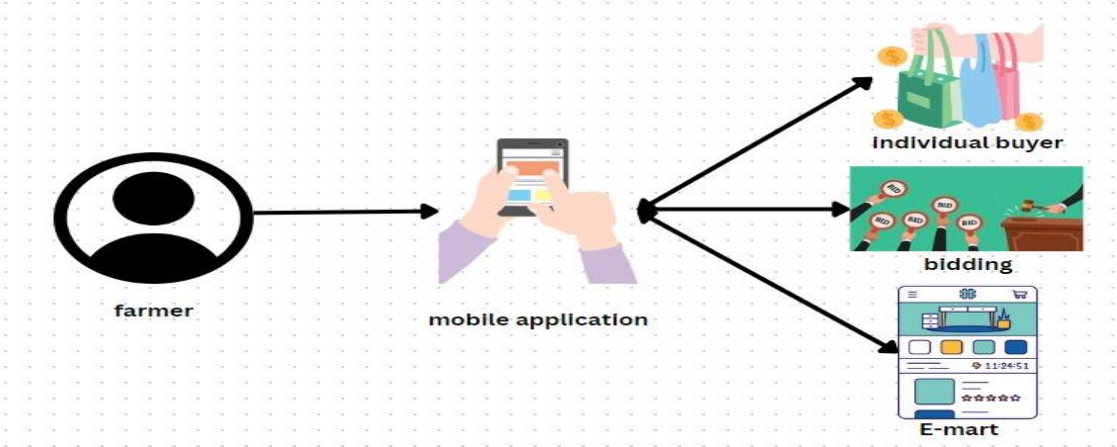


Fig-2.5: Selling Mechanisms

- The system shall offer multiple methods for selling crops, including individual buyer transactions, smart contract-based bidding, and a digital market interface.
- It shall facilitate transparent and efficient crop selling processes, including price negotiation and transaction settlement.

Stubble Utilization:

- The system shall enable the selling of crop residues (stubble) to bioenergy-producing industries.

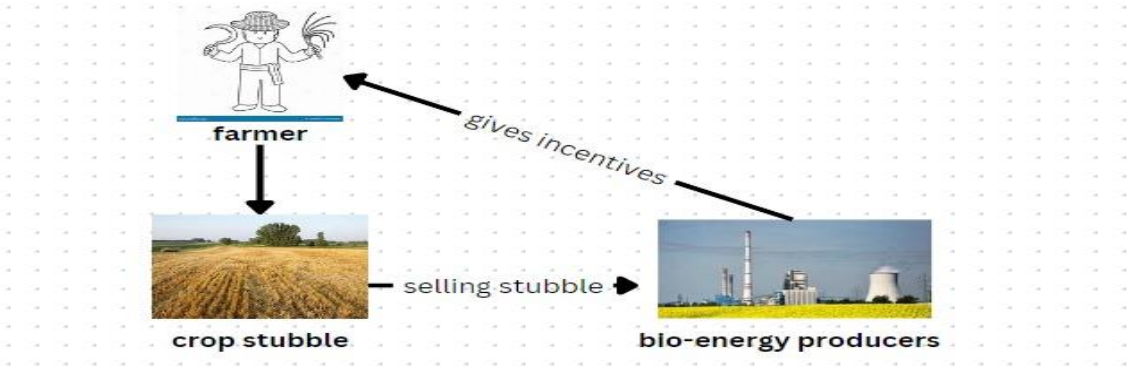


Fig-2.6: Stubble Utilization

- It shall ensure transparent stubble sales transactions and securely store related data in the blockchain for traceability.

Insurance Scheme Management:

Fig-2.7: Scheme Management

- The system shall list various insurance schemes tailored to farmers' needs, providing financial protection against natural calamities.
- It shall enable farmers to securely store final insurance scheme reports in the blockchain for transparency and data integrity.

User Management and Authentication:

Fig-2.8: Authentication

- The system shall allow users to register and create accounts with unique credentials (ID, password).
- It shall authenticate users' credentials during the login process to ensure secure access to the system.

Profile Management:

- The system shall enable users to manage their profiles, including personal information, preferences, and notification settings.
- It shall allow users to update and modify their profiles as needed.

Reporting and Analytics:

- The system shall generate reports and analytics based on collected data, including crop health status, yield predictions, and financial summaries.
- It shall provide visualizations and insights to help users make informed decisions and optimize agricultural practices.

Integration and Compatibility:

- The system shall integrate seamlessly with existing agricultural infrastructure and technologies, including IoT sensors and blockchain platforms.
- It shall be compatible with various devices and platforms, ensuring accessibility and usability for users.

By defining these functional requirements, can ensure that our agricultural management system effectively addresses the needs of users and stakeholders, providing valuable features and functionalities to optimize agricultural practices and improve productivity.

2.7.2 Non-Functional Requirements (NFRs)

Apart from functional requirements, we also have some non-functional requirements. **Non-functional requirements or NFRs** are a set of specifications that describe the system's operation capabilities and constraints and attempt to improve its functionality. These are basically the requirements that outline how well it will operate including things like speed, security, reliability, data integrity, etc. **Non-Functional Requirement** (NFRs) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system.

- **Security:** Implement robust data encryption techniques to protect sensitive information such as farmer data, transaction details, and crop monitoring data from unauthorized access or tampering.

- **Reliability:** Ensure the system operates consistently and accurately under varying conditions, minimizing downtime and errors in data collection, analysis, and transaction processing.
- **Portability:** Design the system to be compatible with various devices and platforms, allowing farmers and stakeholders to access and interact with the system seamlessly regardless of their preferred hardware or operating system.
- **Maintainability:** Develop the system with modular and well-documented code, facilitating easier maintenance, updates, and enhancements as technology evolves or new requirements arise.
- **Privacy:** Implement privacy-preserving measures to safeguard personal and confidential information, adhering to data protection regulations and ensuring farmers' privacy rights are respected throughout the system's lifecycle.
- **Efficiency:** Optimize system performance to minimize resource usage, response times, and energy consumption, ensuring smooth operation even in resource-constrained environments and maximizing the system's overall efficiency.

2.7.3 User requirements

User Requirements are used as the primary input for creating system requirements. These requirements are used for objective and testing of product information.

- Real-time monitoring of soil parameters and atmospheric conditions for informed decision-making.
- Seamless integration of organic disease remedies for effective crop management.
- Transparent and flexible crop selling mechanisms catering to diverse farmer and buyer needs.
- Access to tailored insurance schemes ensuring financial security against crop loss.
- Intuitive digital interfaces facilitating easy interaction and transaction processing for all stakeholders.

2.8. Financial plan for the Development of Product

The financial plan for a product development in emotion recognition by textual tweets using voting classifier (LR-SGD) may include the following items:

- **Research and Development:** includes costs associated with researching existing technology and tools to develop the product.

- **Data Collection and Preprocessing:** includes costs associated with collecting, processing and cleaning the data used to train the model.
- **Model Development and Testing:** includes costs associated with developing, testing, and refining the voting classifier (LR-SGD) model.
- **Server and Infrastructure:** includes costs associated with setting up and maintaining the server and infrastructure necessary to run the model.
- **Marketing and Advertising** includes costs associated with promoting the product to potential customers.
- **Legal and IP Protection:** includes costs associated with protecting intellectual property and obtaining necessary legal clearance.
- **Employee Salaries and Benefits:** includes costs associated with hiring and compensating the team responsible for developing and maintaining the product.
- **Overhead Costs:** includes costs associated with office rent, utilities, and other general business expenses.

It's important to have a realistic and detailed budget for the project, taking into consideration all the necessary expenses and contingencies.

2.9 Business Plan from Seeding to Commercialization

Here's the breakdown of equipment costs for your agricultural management system project:

Computers: ₹1,00,000

Laptops: ₹2,60,000

IoT Sensors: ₹1,50,000

Blockchain Infrastructure: ₹3,00,000

Machine Learning Workstation: ₹2,00,000

Mouse: ₹4,000

Servers for Rent (Duration 6 Months): ₹1,20,000

Domain Registration (6 Months): ₹12,000

Payroll (Development Team): ₹9,50,000

Mentorship Program: ₹2,00,000

Travelling Expenses (for on-site installations and training): ₹1,50,000

Miscellaneous (Office Supplies, Utilities): ₹2,00,000

Maintenance (Hardware and Software): ₹1,50,000

Legal Fees (Patent Filing, Contracts): ₹12,000

Security (Cybersecurity Measures, Data Protection): ₹20,00,000

Marketing (Digital Marketing, Advertising): ₹3,00,000

Rent costs are excluded as the project was established within a college environment, and rent was not paid. These equipment costs cover essential hardware, software, infrastructure, and operational expenses required for the development, deployment, and maintenance of the agricultural management system.

2.9.1 Business Model Canvas

Key Partners	Key Activities	Value Propositions	Customer Relationships	Customer Segments
1. IoT Sensor Manufacturers 2. Blockchain Platform Providers 3. Machine Learning Experts 4. Agricultural Experts	Develop and maintain the software platform for data collection, analysis, and decision-making. Integrate IoT sensors, blockchain technology, and machine learning algorithms into the system. Provide training and support to users for system adoption and usage. Continuously innovate and update the system to incorporate new technologies and features.	1.Optimized Agricultural Practices 2. Transparent and Efficient Transactions 3. Disease Management Solutions: 4. Financial Security	1.Personalized Support 2.Continuous Communication 3.Community Engagement:	1. Farmer 2. Crop Buyers/Stakeholders 3. Bioenergy-producing Industries 4. Insurance Provider.
	Key Resources		**Channels**	
	1. Software Developers 2. Data Scientists 3. Sales and Marketing Team 4. Customer Support Team		1. Online Platform: 2. Sales and Marketing Efforts 3. Partnerships	

Cost Structure	Revenue Streams
1.Development and Maintenance Costs 2.Hardware Costs 3.Marketing and Sales Expenses 4.Personnel Costs:	1. Subscription Model 2. Transaction Fees 3. Consultation Services

Fig-2.9: Business model canvas

Key Partners:

- **IoT Sensor Manufacturers:** Provide essential hardware components for data collection.
- **Blockchain Platform Providers:** Facilitate secure and transparent transactions and data storage.
- **Machine Learning Experts:** Assist in developing algorithms for disease identification and crop management.
- **Agricultural Experts:** Provide domain-specific knowledge and guidance for system development and implementation.

Key Activities:

- Develop and maintain the software platform for data collection, analysis, and decision-making.
- Integrate IoT sensors, blockchain technology, and machine learning algorithms into the system.
- Provide training and support to users for system adoption and usage.
- Continuously innovate and update the system to incorporate new technologies and features.

Key Resources:

- **Software Developers:** Build and maintain the software platform and backend infrastructure.
- **Data Scientists:** Develop machine learning models for disease identification and crop management.
- **Sales and Marketing Team:** Promote the system to farmers, agricultural organizations, and other stakeholders.
- **Customer Support Team:** Provide assistance and troubleshooting for users.

Value Proposition:

- **Optimized Agricultural Practices:** Enhance crop quality, yield, and resource utilization through real-time data monitoring and analysis.
- **Transparent and Efficient Transactions:** Facilitate transparent crop selling mechanisms and stubble utilization transactions using blockchain technology.
- **Disease Management Solutions:** Detect crop diseases early and provide organic remedy recommendations for effective disease management.

- **Financial Security:** Offer insurance schemes tailored to farmers' needs, providing financial protection against natural calamities.

Customer Segments:

- **Farmers:** Primary users of the system, seeking to optimize their agricultural practices and improve productivity.
- **Crop Buyers/Stakeholders:** Interested in transparent and efficient crop selling mechanisms facilitated by the system.
- **Bioenergy-producing Industries:** Seeking to purchase crop residues (stubble) for bioenergy production.
- **Insurance Providers:** Offering tailored insurance schemes to provide financial security to farmers.

Channels:

- **Online Platform:** Provide access to the system through a web-based interface for easy accessibility and usage.
- **Sales and Marketing Efforts:** Promote the system through digital marketing, workshops, and industry events to reach potential users and stakeholders.
- **Partnerships:** Collaborate with agricultural organizations, government agencies, and industry partners to expand reach and adoption.

Customer Relationships:

- **Personalized Support:** Offer personalized assistance, training, and troubleshooting to users to ensure a positive user experience.
- **Continuous Communication:** Maintain regular communication with users to gather feedback, address concerns, and provide updates on system enhancements.
- **Community Engagement:** Foster a community of users and stakeholders through forums, user groups, and social media to share knowledge and best practices.

Revenue Streams:

- **Subscription Model:** Charge users a subscription fee for access to the system's features and functionalities.
- **Transaction Fees:** Earn revenue through transaction fees for crop selling mechanisms and stubble utilization facilitated by the system.
- **Consultation Services:** Offer consulting services, training programs, and customized solutions to users seeking additional support and expertise.

Cost Structure:

Computers: ₹1,00,000

Laptops: ₹2,60,000

IoT Sensors: ₹1,50,000

Blockchain Infrastructure: ₹3,00,000

Machine Learning Workstation: ₹2,00,000

Mouse: ₹4,000

Servers for Rent (Duration 6 Months): ₹1,20,000

Domain Registration (6 Months): ₹12,000

Payroll (Development Team): ₹9,50,000

Mentorship Program: ₹2,00,000

Travelling Expenses (for on-site installations and training): ₹1,50,000

Miscellaneous (Office Supplies, Utilities): ₹2,00,000

Maintenance (Hardware and Software): ₹1,50,000

Legal Fees (Patent Filing, Contracts): ₹12,000

Security (Cybersecurity Measures, Data Protection): ₹20,00,000

Marketing (Digital Marketing, Advertising): ₹3,00,000

These equipment costs cover essential hardware, software, infrastructure, and operational expenses required for the development, deployment, and maintenance of the agricultural management system.

CHAPTER – 3: DESIGN

3.1. Design concepts & Constraints

Design Concepts

The set of fundamental software design concepts are as follows:

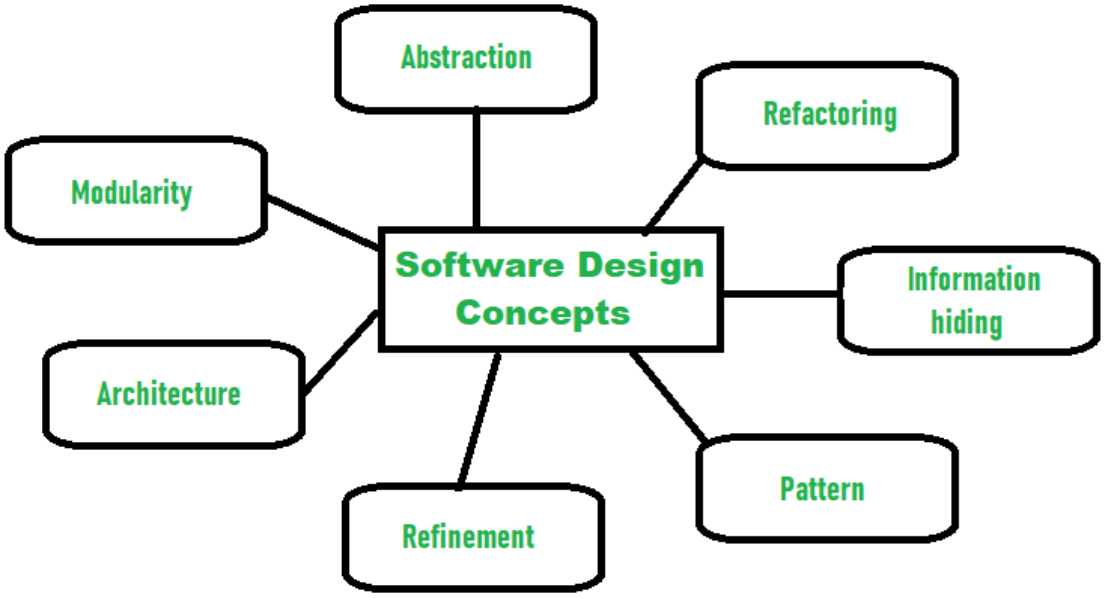


Fig-3.1: Design Concepts

1. Abstraction

A solution is stated in large terms using the language of the problem environment at the highest-level abstraction. The lower level of abstraction provides a more detail description of the solution. A collection of data that describes a data object is a data abstraction.

2. Architecture

The complex structure of the software is known as software architecture. Structure provides conceptual integrity for a system in a number of ways and the architecture is nothing but the structure of program modules where they interact with each other in a specialized way.

3. Patterns

A design pattern describes a design structure and that structure solves a particular design problem in a specified content. Design patterns helps to solve a particular design problem.

4. Modularity

Software is separately divided into name and addressable components and sometimes they are called as modules which integrate to satisfy the problem requirements. It is a single attribute of software that permits a program to be managed easily.

5. Information hiding

Modules must be specified and designed so that the information like algorithm and data in a module is not accessible for other modules not requiring that information.

6. Functional independence

The functional independence is the concept of separation and related to the concepts of modularity, abstraction and information hiding.

It can be accessed by using two criteria

i) Cohesion

A cohesive module performs a single task and it requires a small interaction with the other components in other parts of the program.

ii) Coupling

Coupling is an indication of interconnection between modules in a structure of software.

7. Refinement

A hierarchy is established by decomposing a statement of function in a stepwise manner till the programming language statements are reached. And it follows top-down design approach.

8. Refactoring

Refactoring is the process of changing the software system in a way that it does not change the external behavior of the code but still improves the internal structure. It simplifies the design components without changing its function behavior.

9. Design classes

The model of software is defined as a set of design classes. Every class describes the elements of problem domain and that focus on features of the problem which are user visible.

3.2 Logical Design

A logical design is a conceptual, abstract design. You do not deal with the physical implementation details yet; you deal only with defining the types of information that you need. The process of logical design involves arranging data into a series of logical relationships called entities and attributes. An entity represents a chunk of information. In relational databases, an entity often maps to a table. An attribute is a component of an entity and helps define the uniqueness of the entity. In relational databases, an attribute maps to

a column. You can create the logical design using a pen and paper, or you can use a design tool such as Oracle Warehouse Builder or Oracle Designer While entity relationship diagramming has traditionally been associated with highly normalized models such as online transaction processing (OLTP) applications, the techniques still useful in dimensional modeling.

Conceptual Design

Conceptual design is the very first phase of design, in which drawings or solid models are the dominant tools and products. The conceptual design phase provides a description of the proposed system in terms of a set of integrated ideas and concepts about what it should do, how it should behave, and what it should look like, which will be understandable by users in the manner intended. Visualizing your product is an important aspect of verifying the complete design intent. The primary outputs are the conceptual design(s), design specifications, project schedule, cost estimate, design review, and required Proposal Report Competitive products are additional sources of information for conceptual design.

Logical Design diagrams :

Use case Diagram:

A use case illustrates a unit of functionality provided by the system. The main purpose of the use-case diagram is to help development teams visualize the functional requirements of a system, including the relationship of "actors" (human beings) to essential processes, as well as the relationships among different use cases.

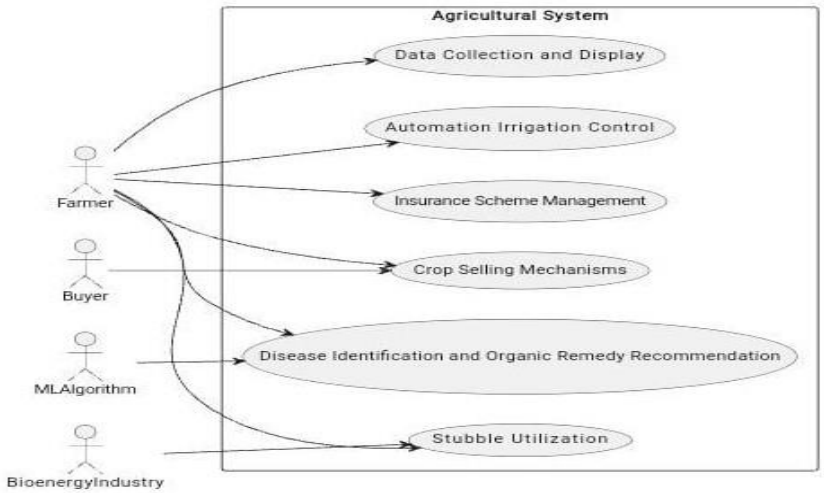


Fig-3.2: Usecase Diagram

Sequence Diagram:

Sequence diagrams show a detailed flow for a specific use case or even just part of a specific use case. They are almost self- explanatory, they show the calls between the 27 different objects in their sequence and can show, at a detailed level, different calls to different objects. A sequence diagram has two dimensions: The vertical dimension shows the sequence of messages/calls in the time order that they occur; the horizontal dimension shows the object instances to which the messages are sent.

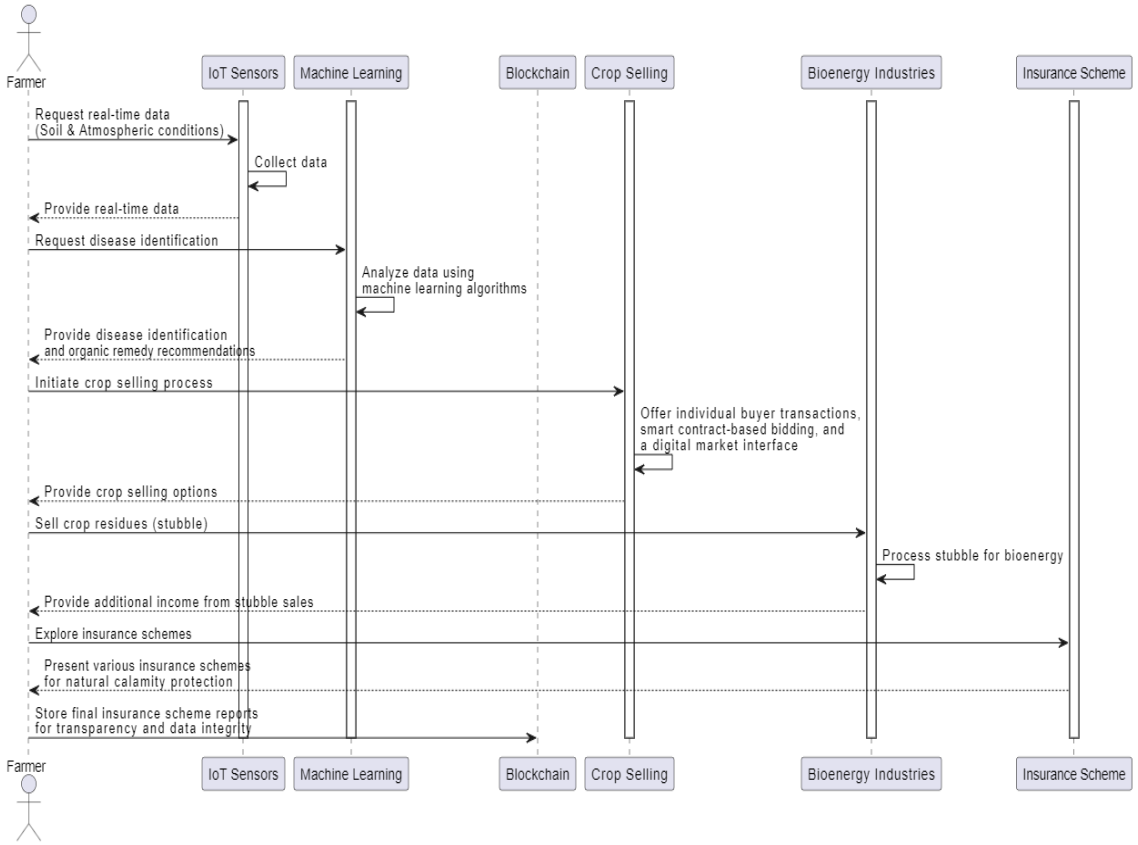


Fig-3.3: Sequence Diagram

Activity Diagram:

Activity diagrams show the procedural flow of control between two or more class objects while processing an activity. Activity diagrams can be used to model higher level business process at the business unit level, or to model low-level internal class actions. In my experience, activity diagrams are best used to model higher-level processes, such as how the company is currently doing business, or how it would like to do business in future.

Fig-3.4: Activity Diagram

Collaboration diagram

A Collaboration diagram models the interactions between objects or parts in terms of sequenced messages. Communication diagrams represent a combination of information taken from Class, Sequence, and Use Case Diagrams describing both the static structure and dynamic behaviour of a system.

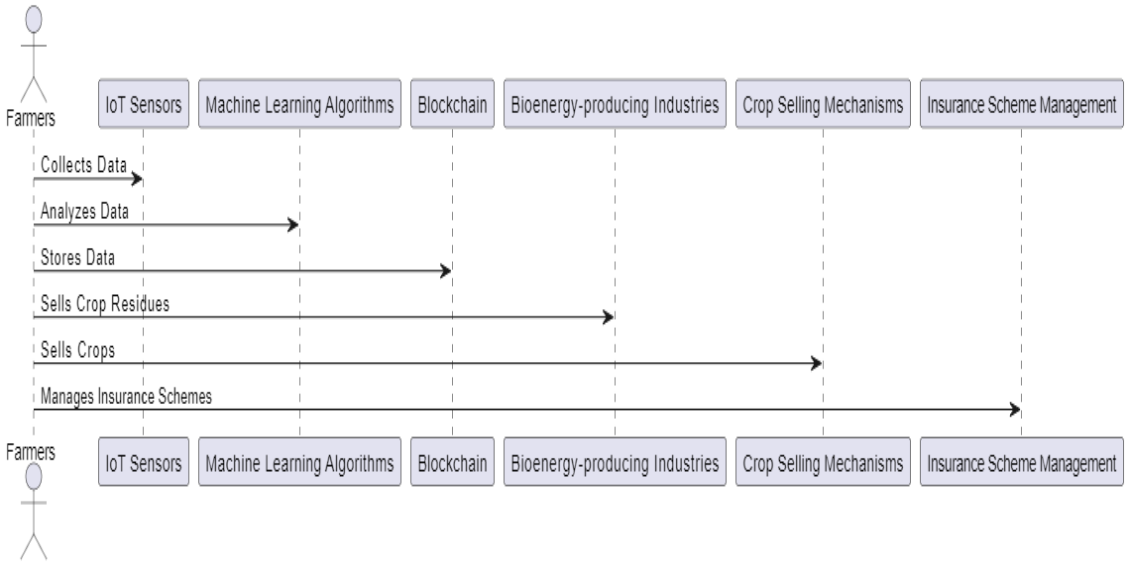


Fig-3.5: Collaboration Diagram

Dataflow Diagram:

Flow diagram is a collective term for a diagram representing a flow or set of dynamic relationships in a system. It is represented in Figure 6.

A data flow diagram (DFD) is a way of representing a flow of a data of a process or a system, usually an information system. The DFD also provides information about the outputs and inputs of each entity and the process itself.

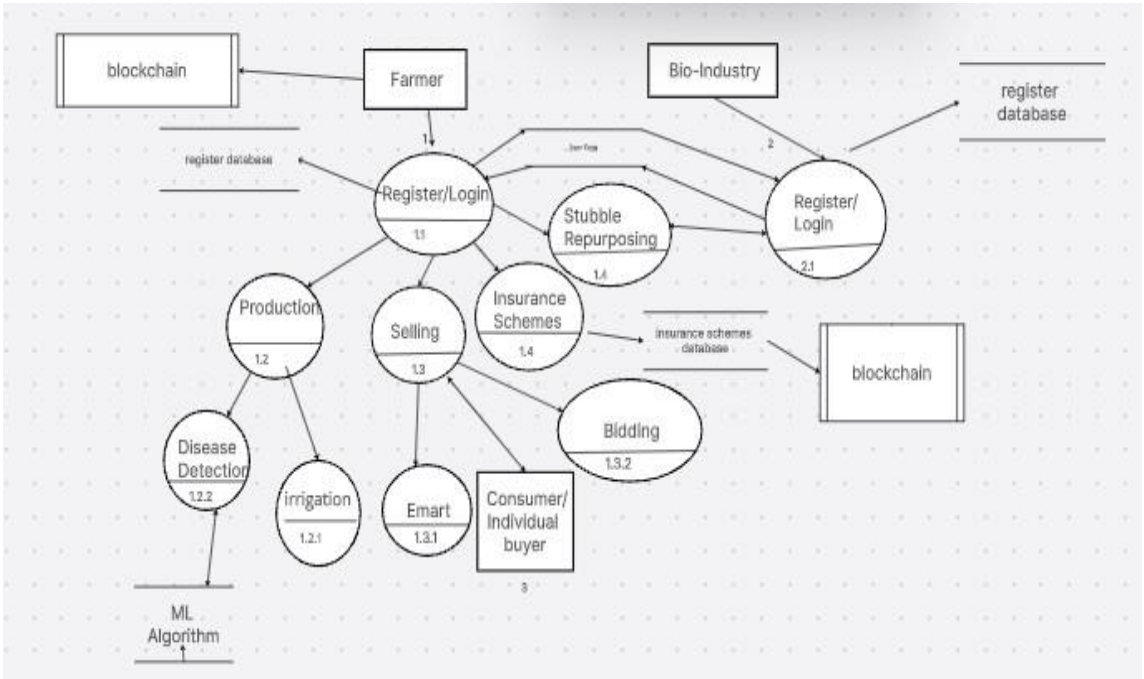


Fig-3.6: Dataflow Diagram

3.3 Architectural Design

Architectural view of entire proposed system is shown in the below figure 3.3.

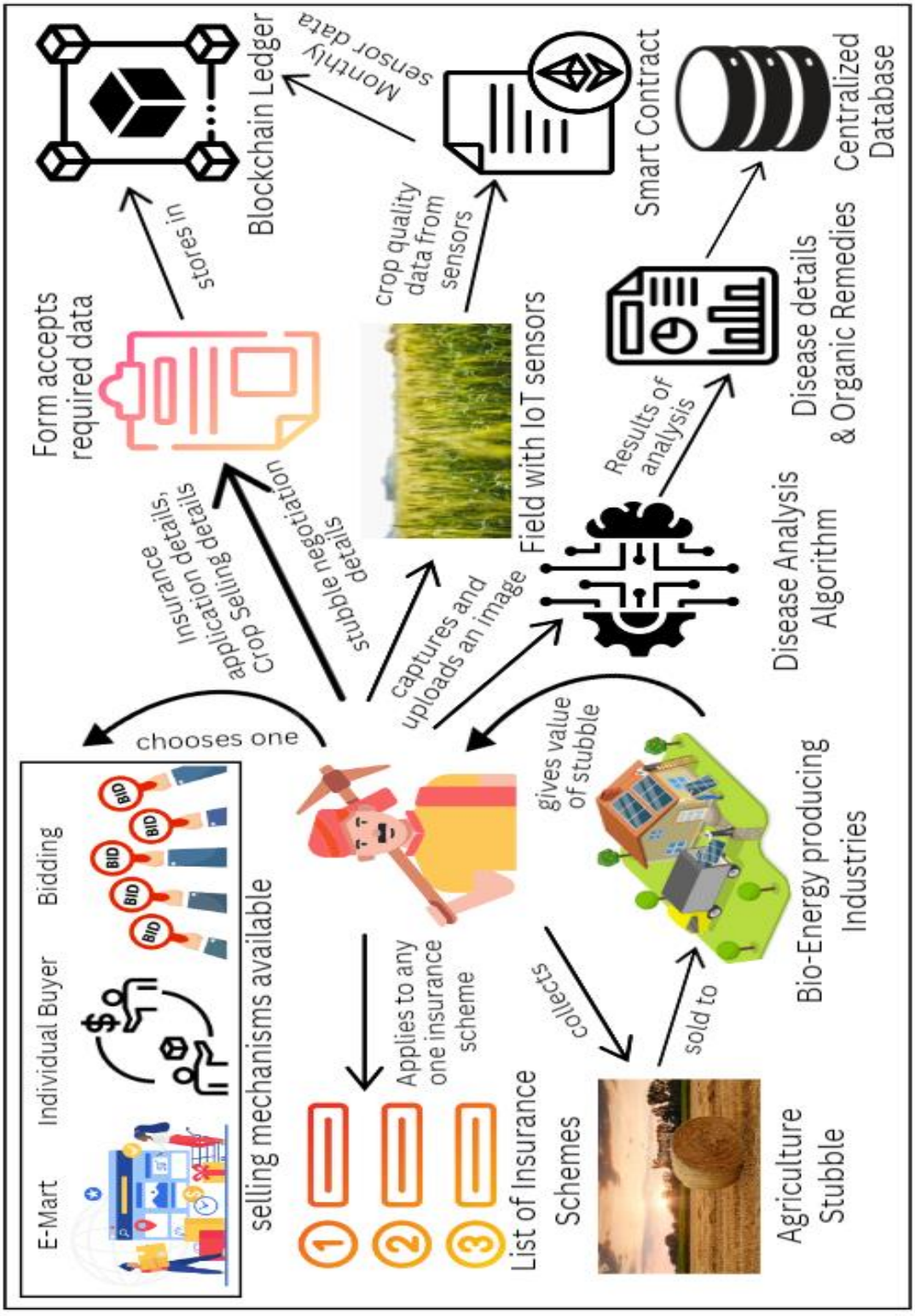


Fig-3.7: Architecture diagram

1. **Register and Login:**

- Allows users, including farmers and stakeholders, to create accounts securely.
- Manages user authentication and authorization processes for accessing the system's functionalities.
- Ensures data privacy and security through encrypted login credentials and authentication protocols.

2. Data Collection:

- Utilizes IoT sensors to collect real-time data on soil parameters and atmospheric conditions.
- Transmits collected data to the central system for analysis and visualization.

3. Crop Monitoring and Disease Identification:

- Analyzes collected data using machine learning algorithms to monitor crop health and detect diseases.
- Provides farmers with timely alerts and recommendations for organic remedies.

4. Irrigation Control:

- Utilizes moisture data to automate irrigation systems, optimizing water usage and crop health.
- Allows farmers to remotely control and monitor irrigation processes through a user-friendly interface.

5. Crop Selling:

- Offers various selling mechanisms including individual transactions, smart contracts, and a digital market interface.
- Facilitates transparent and efficient transactions between farmers and buyers.

6. Stubble Utilization:

- Manages stubble sales transactions with bioenergy-producing industries, providing farmers with an additional source of income.
- Ensures transparency and securely logs stubble-related data on the blockchain.

7. Insurance Scheme Management:

- Lists and manages tailored insurance schemes for farmers' financial protection against crop loss.
- Stores insurance-related data securely on the blockchain for transparency and integrity.

3.4 Algorithms Design

Algorithms are very important in computer Science. The best chosen algorithm makes sure the computer will do the given task in the best possible manner. In cases where efficiency matters a proper algorithm is really vital to be used. An algorithm is important in optimizing a computer program according to the available resources. Decision tree algorithms which take experiences of previous users and then build a model and if a new user enters his requirements then the decision tree will predict the best location based on his given input. Decision trees don‘t need new users' past experience data. To implement a decision tree model we need to have a dataset and this dataset sometimes will have empty or garbage values and this values will put a bad effect on the decision tree model so we can remove such empty or garbage values by applying pre-process techniques. Sometimes to predict or build a model no need to use all columns (attributes) values from the dataset and these unnecessary attributes can be removed by applying features selection Algorithms.

1. Start
2. User Authentication:
   - If the user is not an existing user:
     - Prompt the user to create an account by providing their name, email ID, and password.
     - Save the account details in the database.
   - Else, if the user is an existing user:
     - Prompt the user to log in by providing their ID and password.
     - Authenticate the user's credentials against the database.
3. Present User Options:
   - Present the user with options:
     - Home
     - Production
     - Selling
     - Stubble
     - Insurance
     - Profile
4. Action Selection:
   - Based on the user's selection, navigate to the corresponding module.
5. Home Details Module:
   - If the user chooses to enter home details:

- Retrieve and display agriculture products selling shop details from the database.

6. Production Phase Module:
   - If the user chooses the production phase:
     - Sub-options:
       - Sensor Data
       - Disease Detection
       - Irrigation
7. Selling Phase Module:
   - If the user chooses the selling phase:
     - Sub-options:
       - Individual Buyer
       - Bidding
       - Emart Application
8. Stubble Module:
   - If the user chooses the stubble option:
     - Retrieve and display bio-industry details from the database.
     - Allow the user to send requests.
9. Insurance Schemes Module:
   - If the user chooses the insurance schemes option:
     - Display insurance schemes details retrieved from the database.
10. Profile Module:
    - If the user chooses the profile option:
      - Retrieve and display personal data from the database.
11. STOP

3.5 Database Design

MongoDB is a popular NoSQL database that utilizes a document-oriented data model. Unlike traditional relational databases, MongoDB stores data in flexible, JSON-like documents, making it highly scalable and adaptable to evolving schemas. Its flexible structure allows for dynamic and hierarchical data storage, which is particularly advantageous for applications with rapidly changing requirements or large volumes of unstructured data. Additionally, MongoDB offers powerful querying capabilities, including support for complex queries, indexing, and aggregation pipelines, making it suitable for a wide range of use cases, from web applications to real-time analytics. Its

robust features, scalability, and ease of use have made MongoDB a popular choice for modern application development.

Table that stores the details of users who register for the application.

Column Name	Condition	Datatype
userName	NOTNULL	VARCHAR2
age	NOTNULL	NUMBER
location	NOTNULL	VARCHAR2
landmark	NOTNULL	VARCHAR2
PhoneNum	PRIMARYKEY	NUMBER(10)
Crop	NOTNULL	VARCHAR2
yearsOfFarming	NOTNULL	NUMBER

Table-3.1: User Details

Table that stores the dealers which sell all the required capital to start a crop production.

Column Name	Condition	Datatype
ShopName	NOTNULL	VARCHAR2
location	NOTNULL	VARCHAR2
PhoneNum	PRIMARYKEY	NUMBER(10)

Table-3.2: Agro Input Dealers Details

Table to store the sensor values.

Column Name	Condition	Datatype
sensorName	PRIMARYKEY	VARCHAR2
value	NOTNULL	REAL(M,D)

Table-3.3: Sensor data

Table to store request details.

Column Name	Condition	Datatype
diseaseName	PRIMARYKEY	VARCHAR2
description		VARCHAR2
remedies	NOTNULL	VARCHAR2

Table-3.4: Requests

Table to store individual buyers details.

Column Name	Condition	Datatype
userName	NOTNULL	VARCHAR2
location	NOTNULL	VARCHAR2
PhoneNum	PRIMARYKEY	NUMBER(10)

Table-3.5: Buyers

Table to store individual buyers details.

Column Name	Condition	Datatype
userName	NOTNULL	VARCHAR2
industryName	NOTNULL	VARCHAR2
location	NOTNULL	VARCHAR2
PhoneNum	PRIMARYKEY	NUMBER(10)

Table-3.6: Bio-Energy producers

CHAPTER – 4: CODING & OUTPUT SCREENS

4.1 Output Screens

Output screens in project documentation refer to the visual representations of the user interface of a software application or system. These screens are typically included in the project documentation to help stakeholders understand the functionality and design of the application or system.

Figure showing the registration page for farmer from where he can enter the details to login to the website.

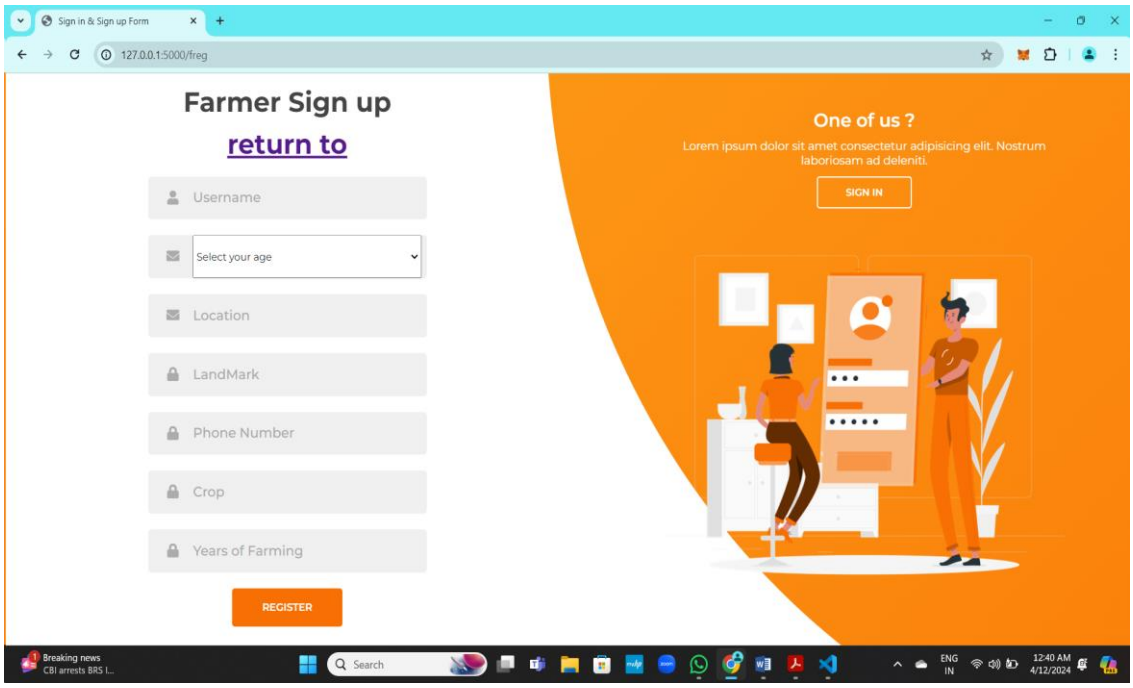


Fig-4.1: Farmer registration page

After successful registration he can login to the website by giving his username and phone number.

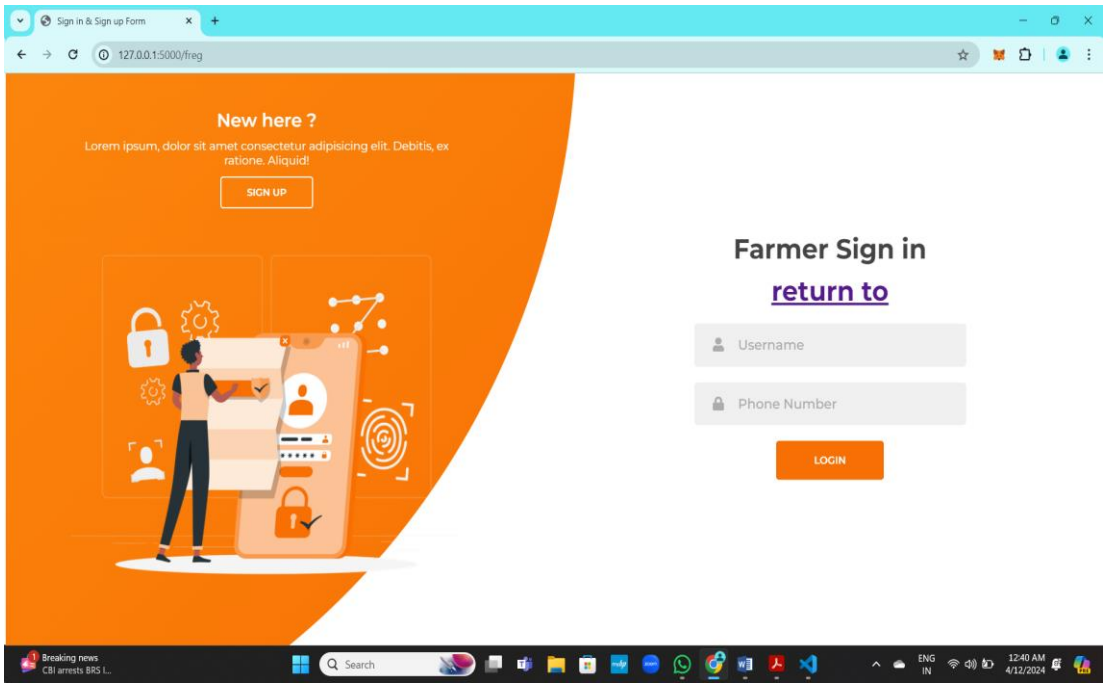


Fig-4.2: Farmer Login page

After successful login he will be navigated to the home page where he can do no.of activities.

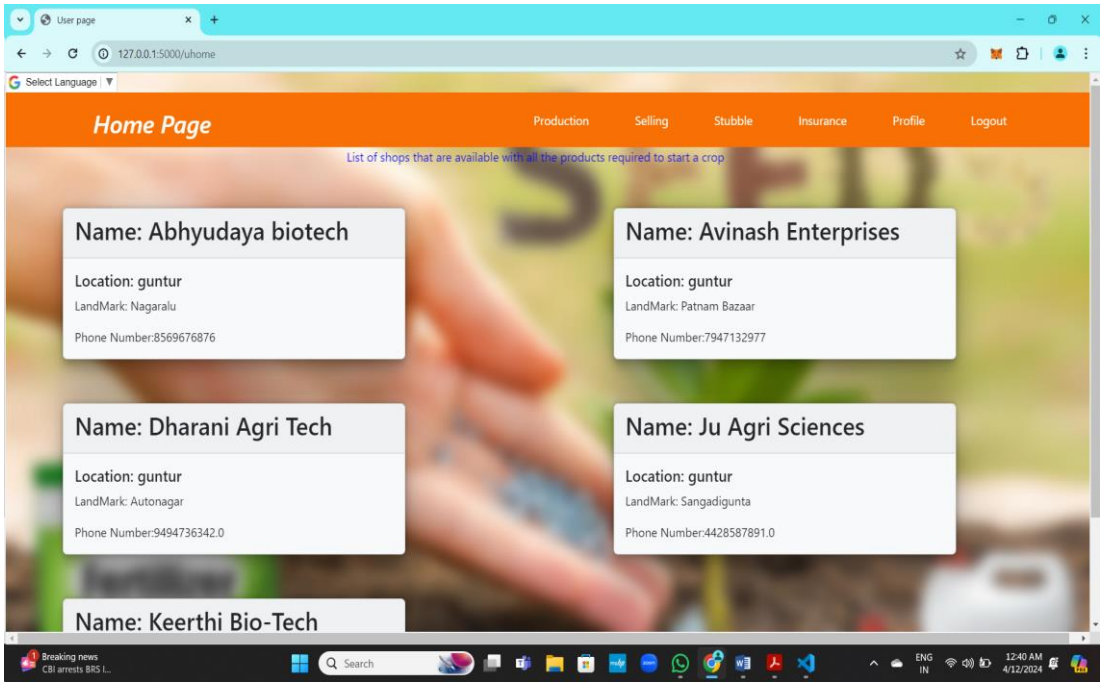


Fig-4.3: Farmer Home page

When he click the production tab he can be landed in the below page.

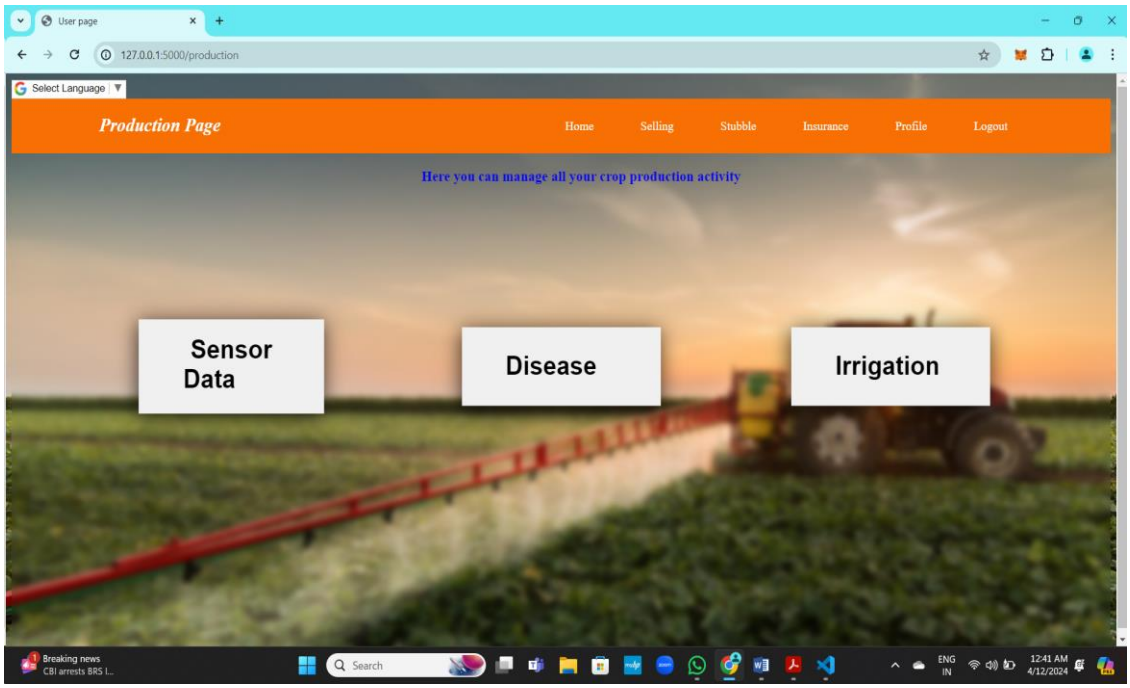


Fig-4.4: Production page

When he click disease detection tab he will be asked to insert the image of the diseased leaf inturn gives the details and remedies.

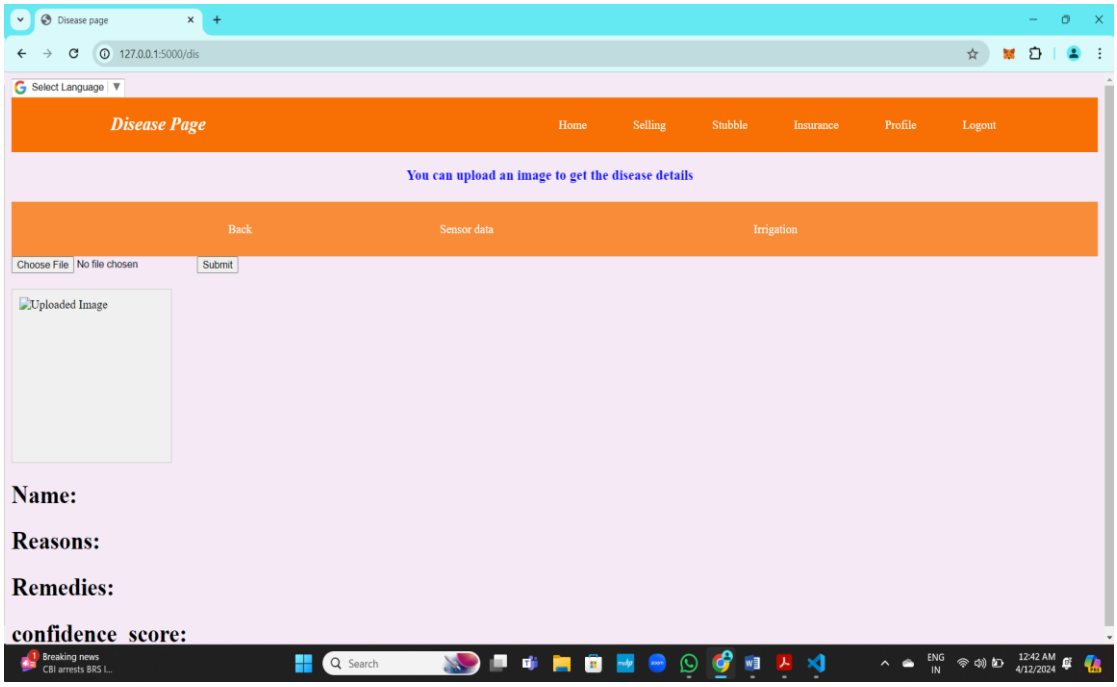


Fig-4.5: Disease Detection page

If he click on irrigation tab he can be able to see the motor activity controlled by moisture data from sensor.

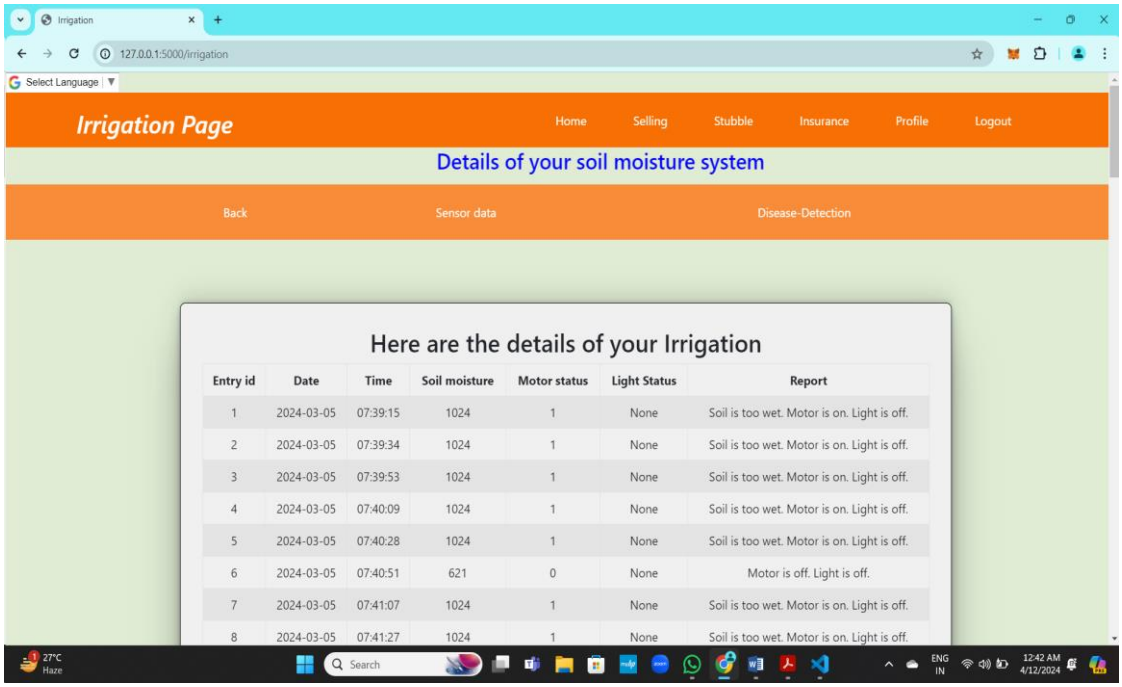


Fig-4.6: Irrigation page

If he click on selling he will be landed in the below page.

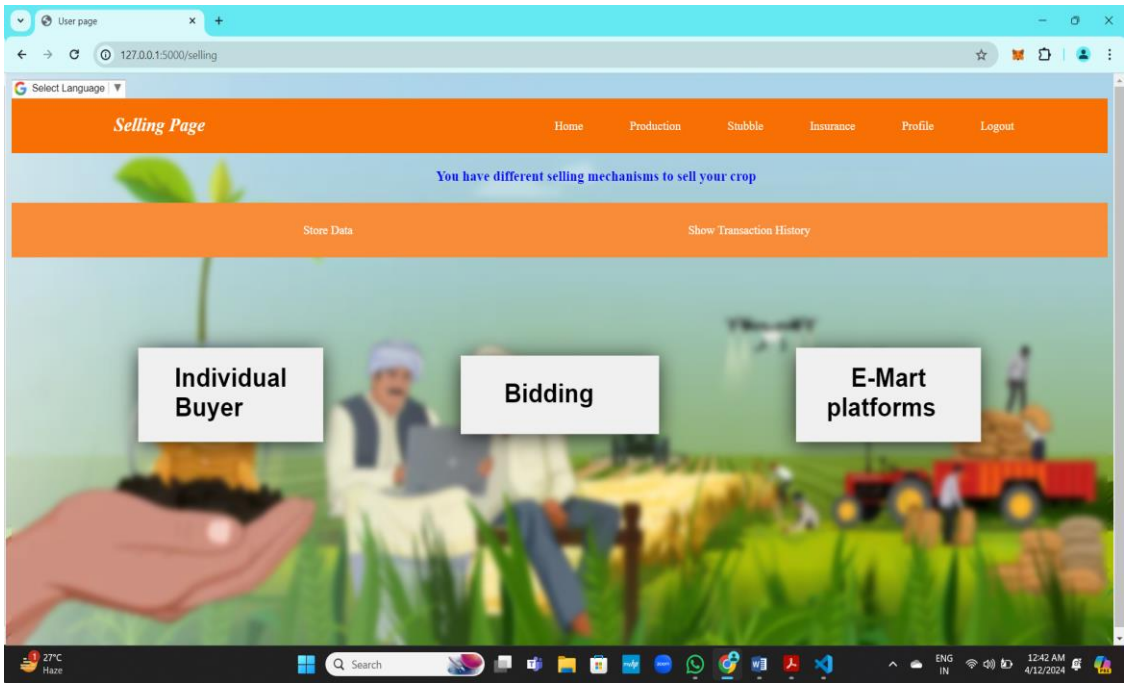


Fig-4.7: Selling page

When he click on individual buyer tab he will be shown the below page.

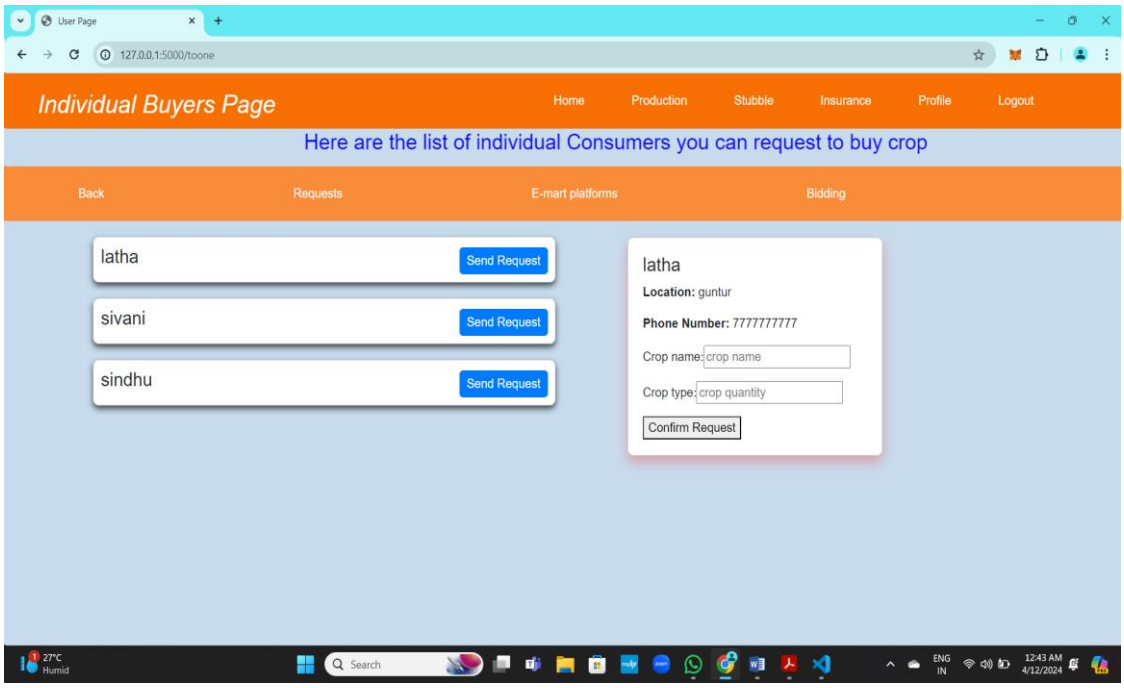


Fig-4.8: Individual Buyer page

If he clicks on emarts tab the below page is shown.

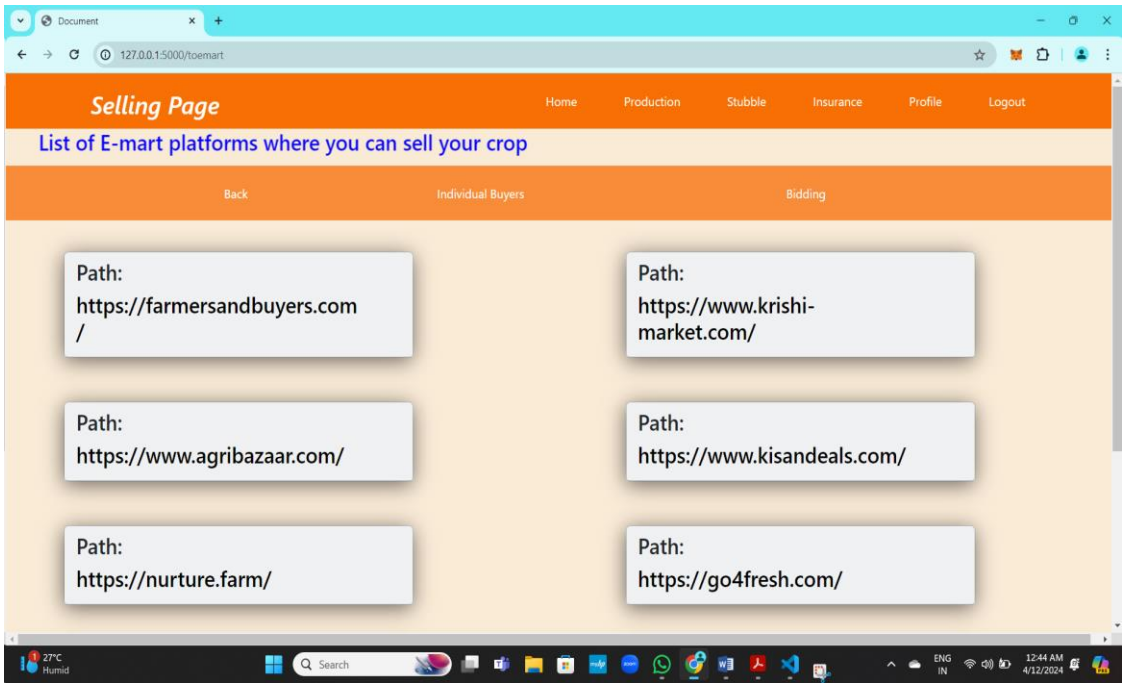


Fig-4.9: E-Marts page

If he clicks on bidding tab he can see the list of bidding platforms.

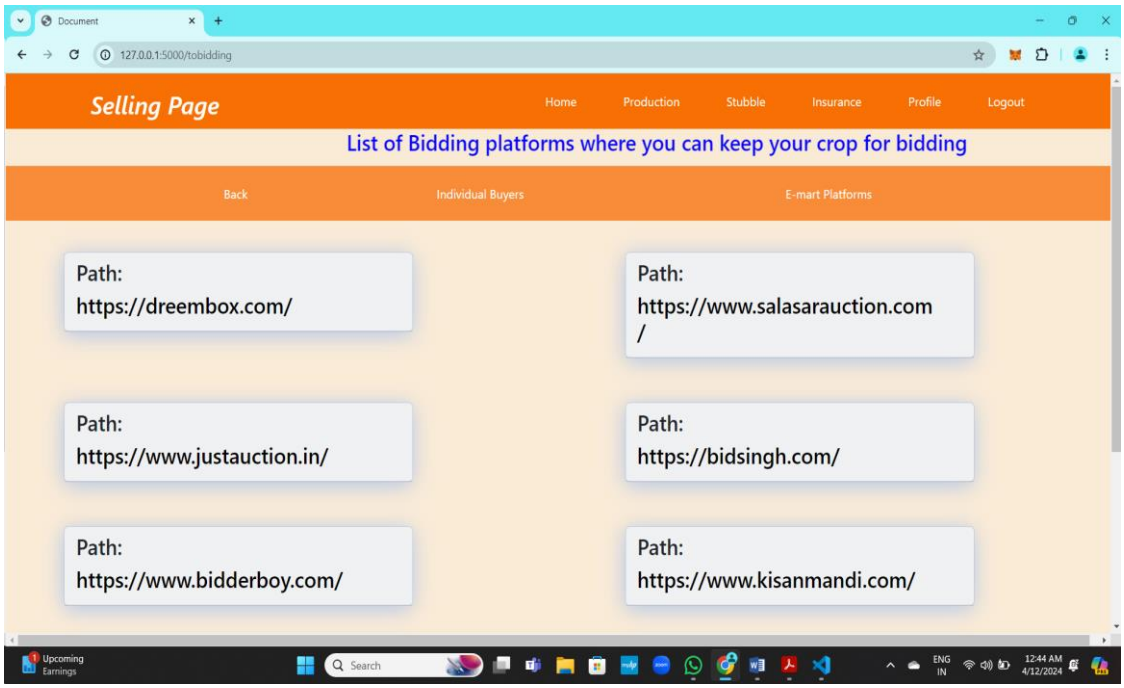


Fig-4.10: Bidding page

He will be navigated to below page if he clicks on stubble and can make requests to the particular industry.

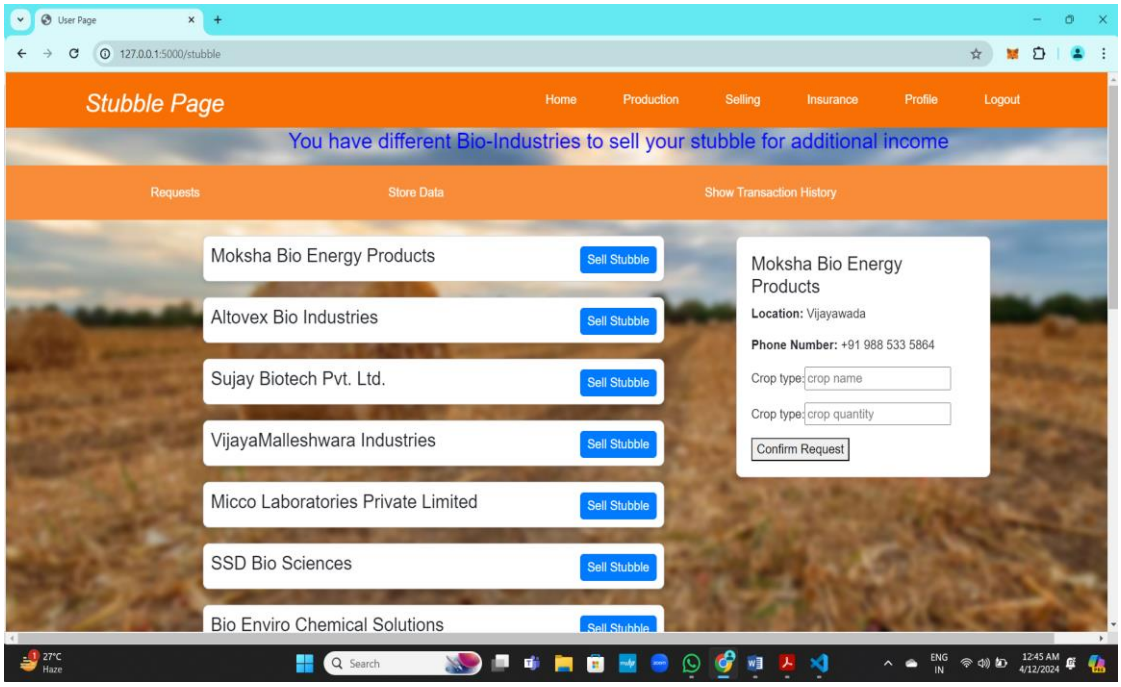


Fig-4.11: Stubble page

If he clicks on insurance he will be shown the list of schemes both central and state level.

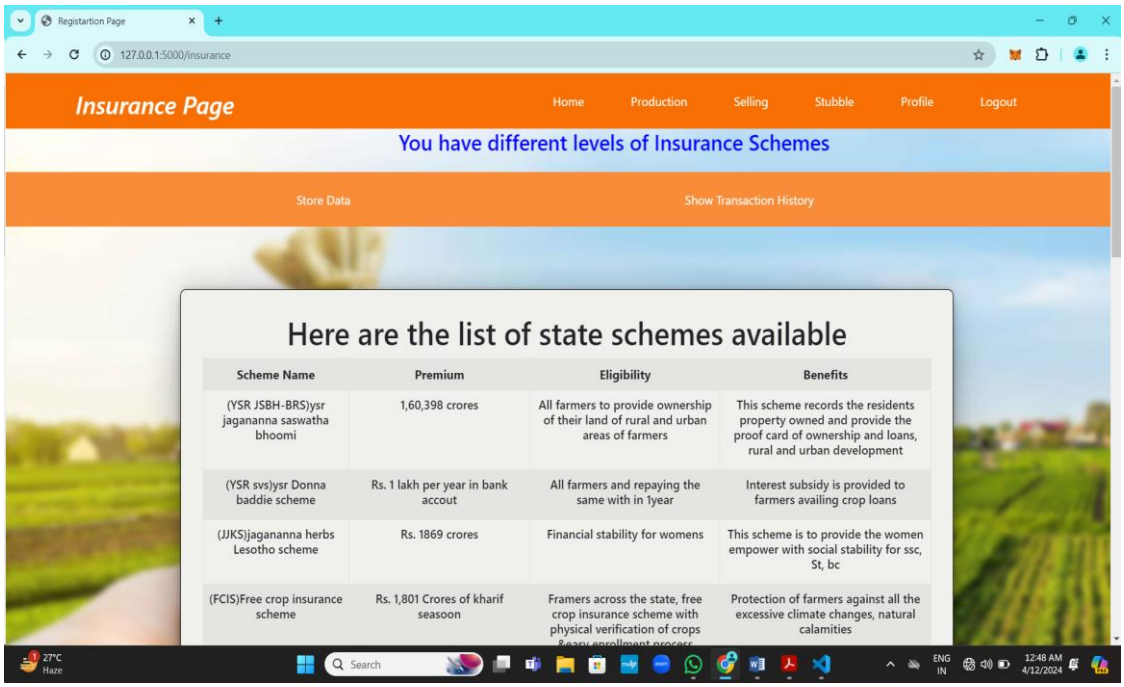


Fig-4.12: Insurance page

If he clicks on profile he will be able to modify the personal details that are given while registering.

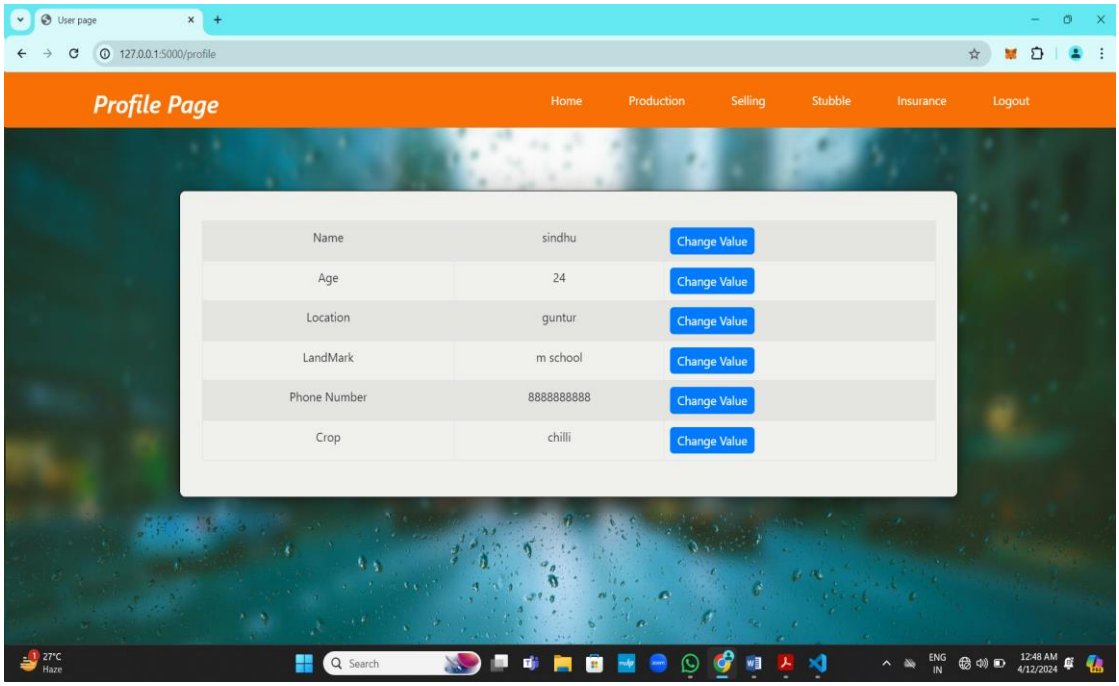


Fig-4.13: Profile page

Figure showing the registration page for Individual consumer from where he can enter the details to login to the website.

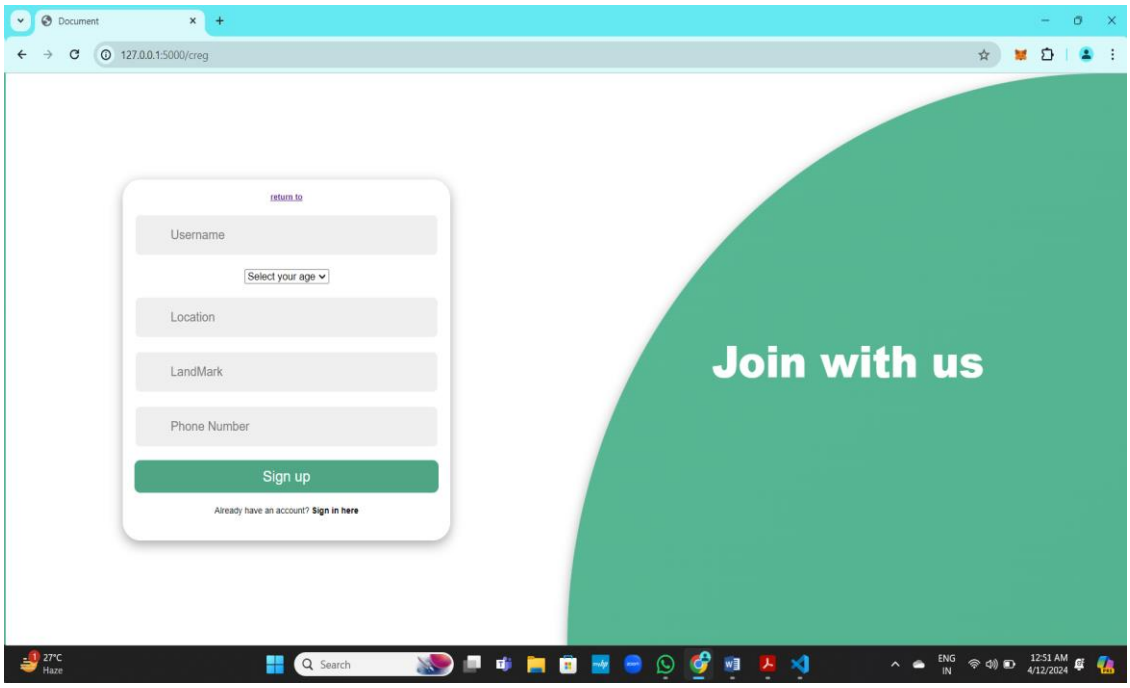


Fig-4.14: Consumer registration page

After successful registration he can login to the website by giving his username and phone number.

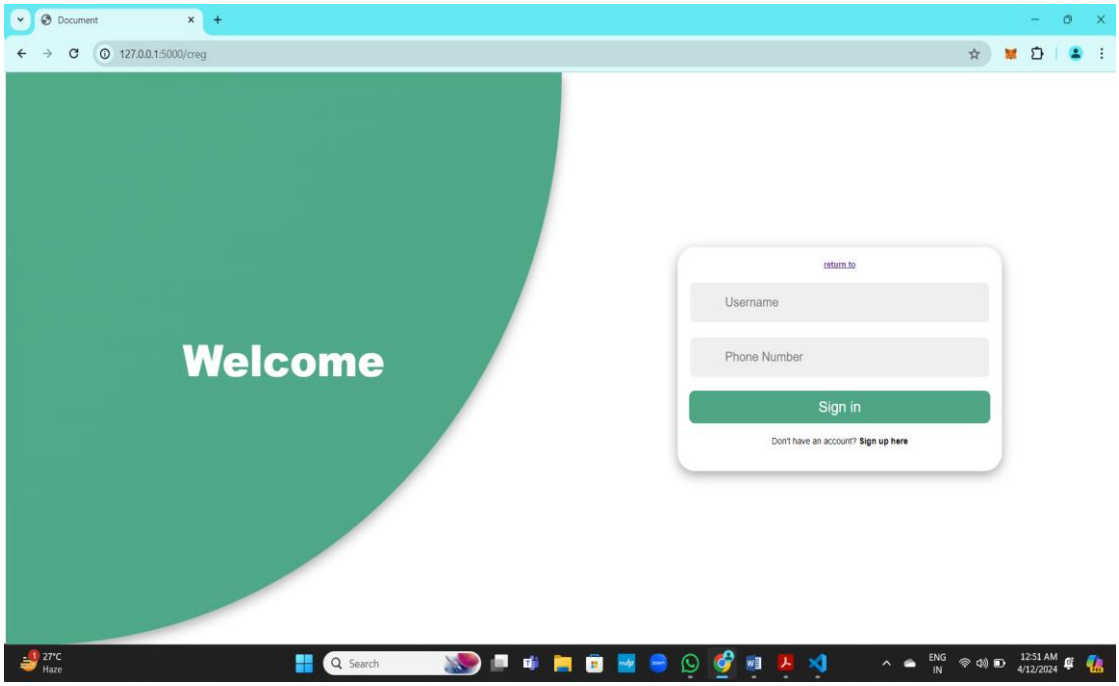


Fig-4.15: Consumer login page

After successful login he will be navigated to the home page where he can see the list of requests he got from the farmer.

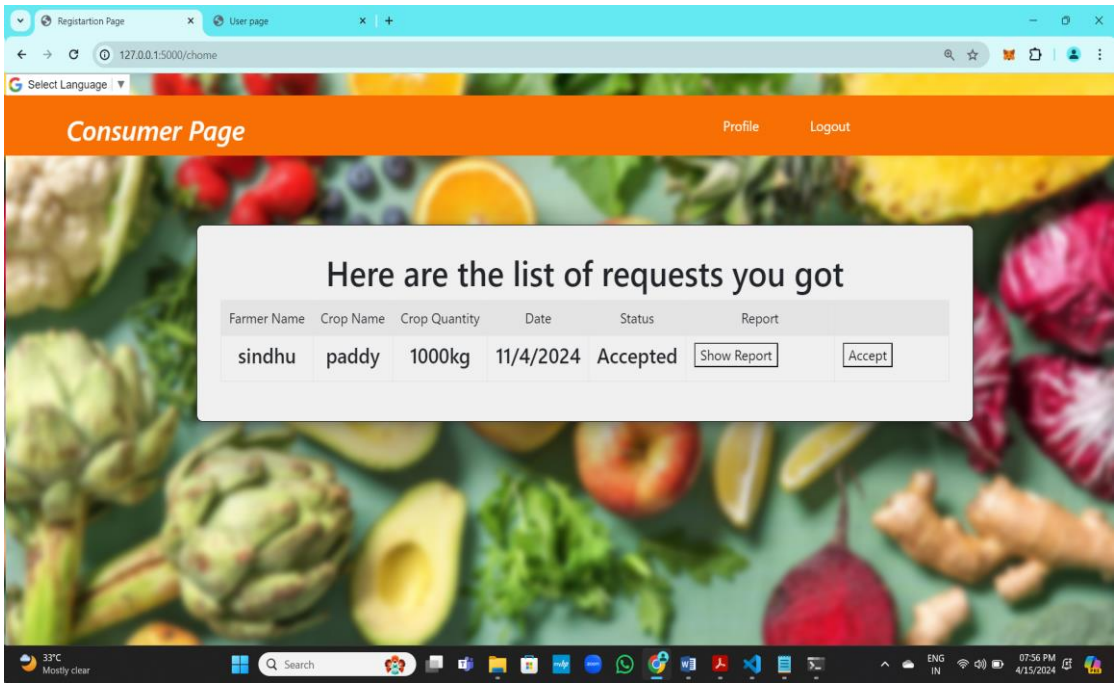


Fig-4.16: Consumer home page

If he click on report he will be shown the crop details.

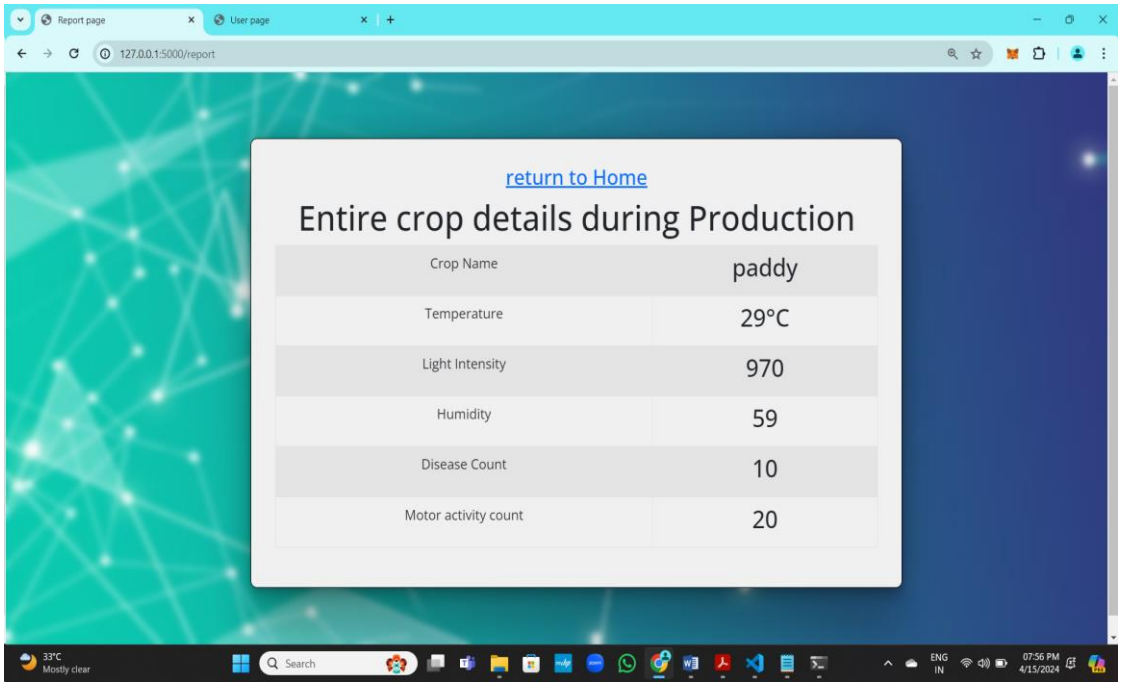


Fig-4.17: Report page

If he clicks on profile he will be able to modify the personal details that are given while registering.

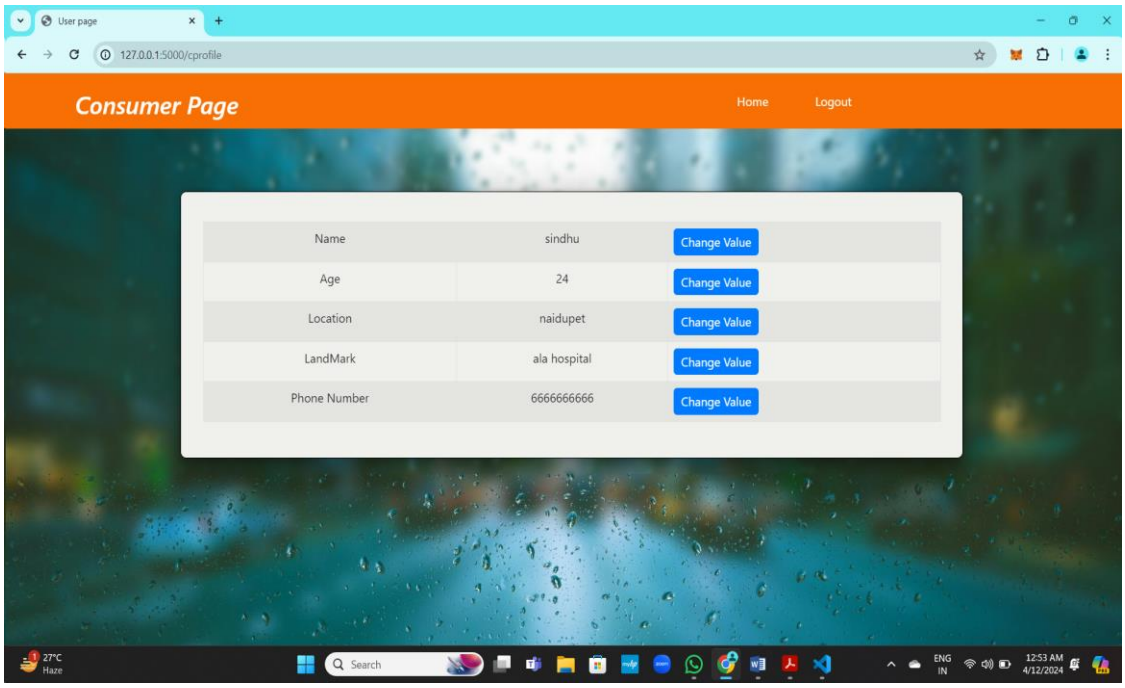


Fig-4.18: Consumer profile page

The bio-industry person can login by selecting the corresponding industry name.

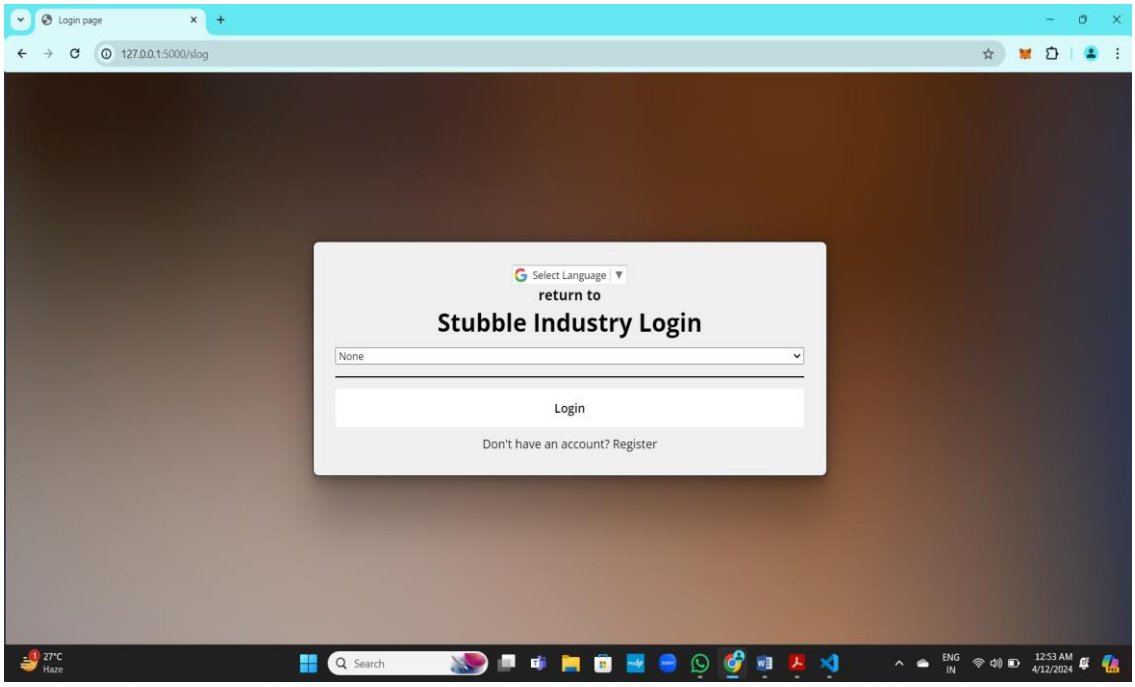


Fig-4.19: Stubble Industry login page

After successful login he will be navigated to the home page where he can see the list of requests he got from the farmer.

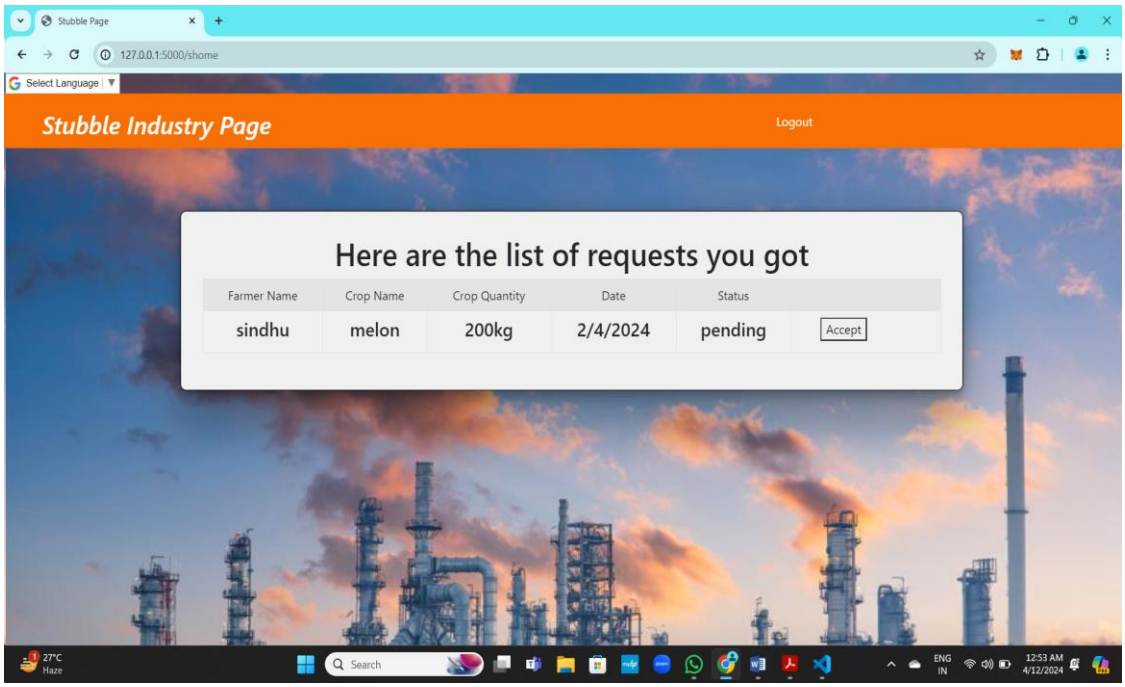


Fig-4.20: Stubble Industry home page

4.2. Screen Reports

A "screen report" is a type of documentation that describes the user interface of a software application or system. It typically includes a visual representation of each screen or page in the user interface, along with a description of the functionality and features of each screen. A screen report is often used as a reference guide for developers, and testers working on the project. It can also be used to communicate the design and functionality of the user interface.

Here are the list of screen reports when the user specific activities can take place.

When he click sensor data tab he will be shown the data collected from sensor include climate parameters.

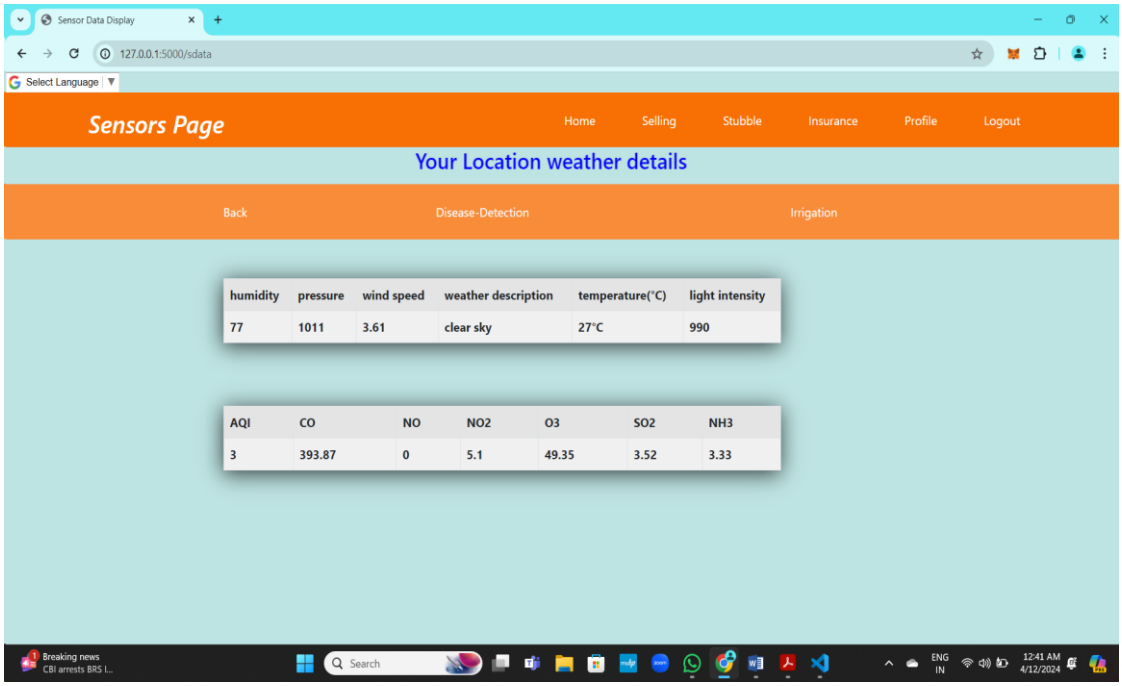


Fig-4.21: Sensor data page

He will be able to see the requests made by himself to the buyers he can also delete the request and can be able to see the status of request.

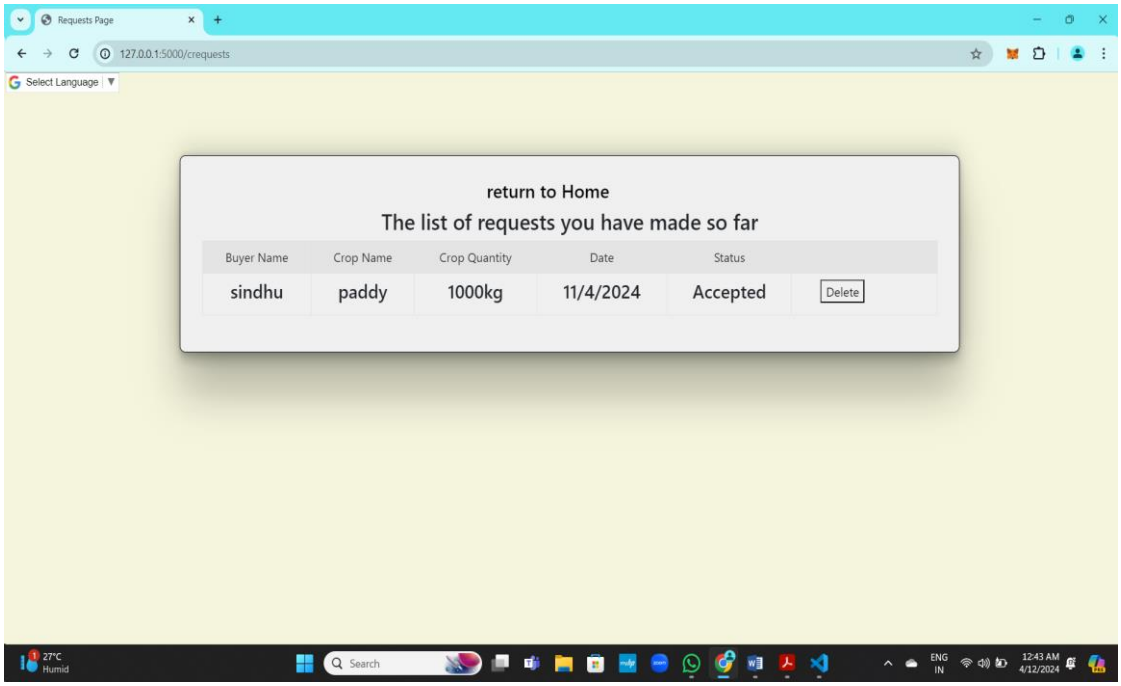


Fig-4.22: Farmer Requests page

He will be able to see the requests made by himself to the bio-industries, he can also delete the request and can be able to see the status of request.

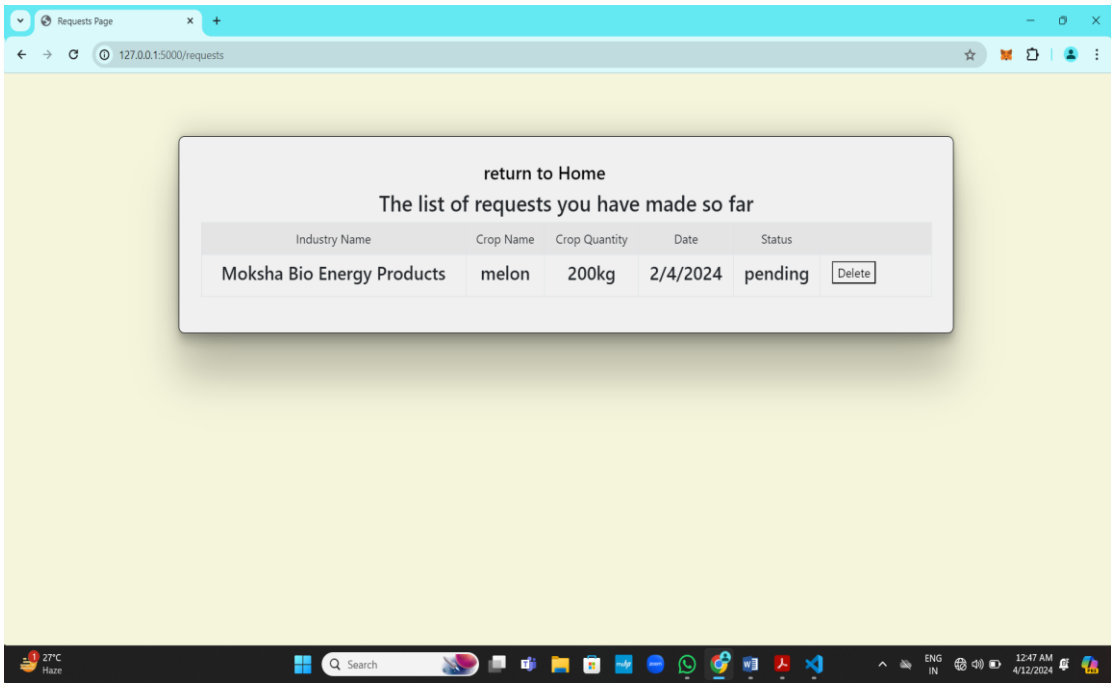


Fig-4.23: Stubble requests page

He can store each and every transaction he has made by entering the required details in the form.

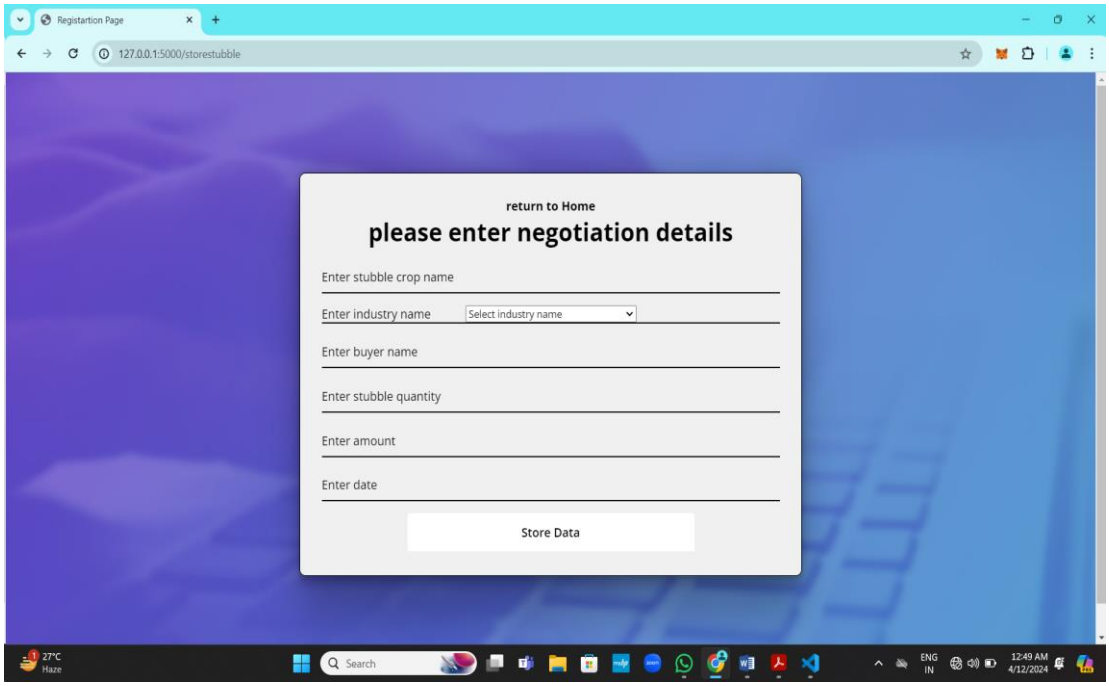


Fig-4.24: Data collection page to store in blockchain

He can also see the details that are stored in the block chain entered by himself.

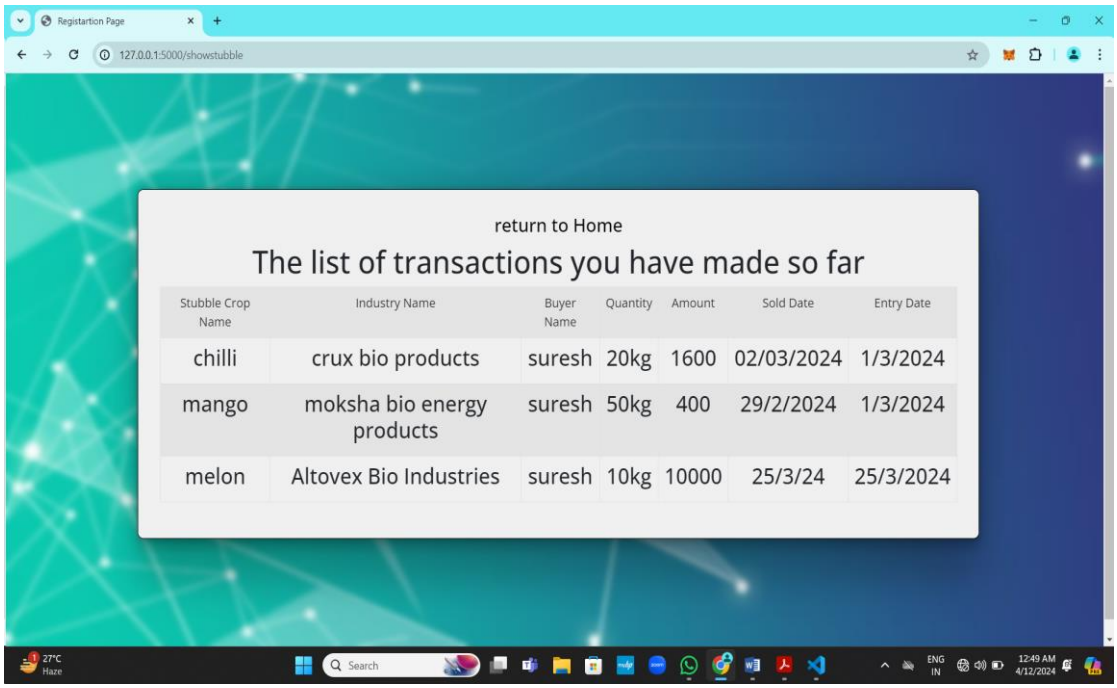


Fig-4.25: Data display page from blockchain

We have already mentioned that negotiation data is stored in block chain that screen will be like this.

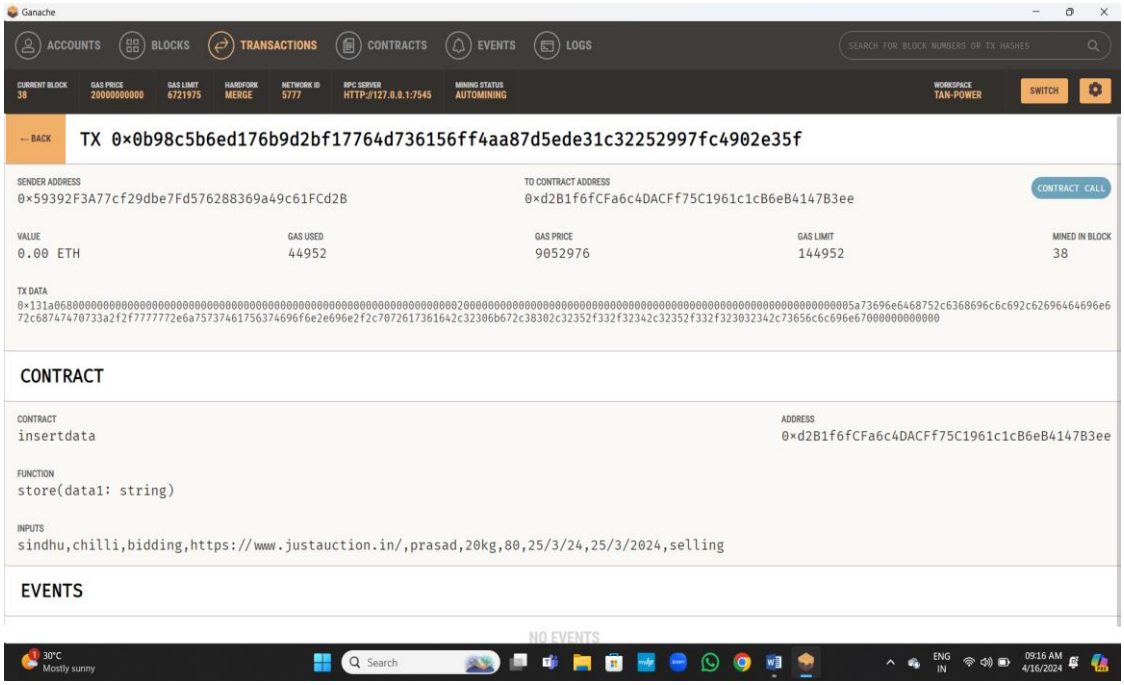


Fig-4.26: Data storage page from blockchain

4.3 Sample Coding

The sample coding phase of an application typically involves writing code that implements the design and functionality of the application based on the specifications and requirements gathered during the planning and design phase.

```
from web3 import Web3,HTTPProvider
import json
import os
from flask import Flask,render_template,request,redirect,session
from bson import ObjectId
from pymongo import MongoClient
from datetime import datetime
from keras.models import load_model
from PIL import Image, ImageOps
import numpy as np
import requests
import urllib3
blockchain="http://127.0.0.1:7545"
web3=Web3(HTTPProvider(blockchain)) #tan-power
app=Flask('name',static_folder='static')
app.secret_key = 'fapo'
dbClient=MongoClient('mongodb://localhost:27017/')
db=dbClient['fapo']
userdata=db['userdata']
agroDealers=db['agroDealers']
indiBuyers=db['indiBuyers']
emarts=db['emarts']
biddings=db['biddings']
bioIndus=db['bioIndus']
negoData=db['negoData']
urequests=db['urequests']
loc=""
phno=""
name=""
```

```
msg1=""
sname=""
crop=""
crop1=""
cname=""
cphno=""
gtemp=""
gli=""
hum=""
@app.route('/')
def home():
    return render_template('home.html')
@app.route('/freg')
def reg():
    return render_template('fregister.html')
@app.route('/creg')
def reg1():
    return render_template('cregister.html')
@app.route('/flog')
def log():
    return render_template('flogin.html')

@app.route('/clog')
def log1():
    return render_template('clogin.html')
@app.route('/slog')
def log2():
    info=bioIndus.find()
    data1=list(info)
    moksha=['Moksha Bio Energy Products','Vijayawada','+91 988 533 5864']
    indus=[]
    indus.append(moksha)
    for i in data1:
```

```
        dummy=[]
        dummy.append(i.get('name'))
        dummy.append(i.get('phno'))
        indus.append(dummy)
    return render_template('slogin.html',data=indus)
@app.route('/production')
def production():
    global loc
    return render_template('production.html')
@app.route('/selling')
def selling():
    return render_template('selling.html')
@app.route('/insurance')
def insurance():
    return render_template('insurance.html')
@app.route('/fregister',methods=['POST','GET'])
def register():
    name=request.form.get('name')
    age=request.form.get('age')
    gender=request.form.get('gender')
    loc=request.form.get('loc')
    landm=request.form.get('landm')
    phno=request.form.get('phno')
    crop=request.form.get('crop')
    yof=request.form.get('yof')
    exis_user=userdata.find_one({'name':name,'phno':phno})
    if exis_user:
        msg="User already existed,Please create new user"
        return render_template('fregister.html',msg=msg)
    else:
      user={'name':name,'age':age,'gender':gender,'loc':loc,'landm':landm,'phno':phno,'cr
op':crop,'yof':yof}
        userdata.insert_one(user)
```

```
        msg="Registartion successful!!!"
        return render_template('fregister.html',msg=msg)
@app.route('/flogin',methods=['POST','GET'])
def login():
    global loc
    global phno
    global name
    global crop
    name=request.form.get('name')
    phno=request.form.get('phno')
    user=userdata.find_one({'name':name,'phno':phno})
    if user!=None:
        loc=user['loc']
        phno=user['phno']
        name=user['name']
        crop=user['crop']
        session['username']=phno
        return redirect('/uhome')
    else:
        msg="Invalid details"
        return render_template('flogin.html',msg=msg)
@app.route('/cregister',methods=['POST','GET'])
def register1():
    name=request.form.get('name')
    age=request.form.get('age')
    gender=request.form.get('gender')
    loc=request.form.get('loc')
    landm=request.form.get('landm')
    phno=request.form.get('phno')
    exis_user=indiBuyers.find_one({'name':name,'phno':phno})
    if exis_user:
        msg="User already existed,Please create new user"
        return render_template('cregister.html',msg=msg)
```

```
    else:
user={'name':name,'age':age,'gender':gender,'loc':loc,'landm':landm,'phno':phno}
        indiBuyers.insert_one(user)
        msg="Registartion successful!!!"
        return render_template('cregister.html',msg=msg)
@app.route('/clogin',methods=['POST','GET'])
def login1():
    global cphno
    global cname
    name=request.form.get('name')
    phno=request.form.get('phno')
    user=indiBuyers.find_one({'name':name,'phno':phno})
    if user!=None:
        cphno=user['phno']
        cname=user['name']
        return redirect('/chome')
    else:
        msg="Invalid details"
        return render_template('clogin.html',msg=msg)
@app.route('/chome',methods=['POST','GET'])
def chome():
    global crop1
    print("name is:",cname)
    info1=urequests.find({'type':'indi','sname':cname})
    info=list(info1)
    # print(info[0]['cname'])
    if len(info)>0:
        crop1=info[0]['cname']
        return render_template('chome.html',data=info)
    else:
        return render_template('chome.html',msg="No requests found!!!")
@app.route('/cupstatus/<x>',methods=['POST','GET'])
def cupstatus(x):
```

```
    name2=x
    status="Accepted"
    urequests.update_one({'_id':ObjectId(name2)},{'$set':{'status':status}})
    return redirect('/chome')


@app.route('/report',methods=['POST','GET'])
def report():
    info=[gtemp,gli,hum,crop1]
    # print(gtemp,gli,hum,crop)
    return render_template('report.html',data=info)
@app.route('/slogin',methods=['POST','GET'])
def login2():
    global sname
    sname=request.form.get('sname')
    return redirect('/shome')
@app.route('/shome',methods=['POST','GET'])
def shome():
    info1=urequests.find({'type':'stubble','sname':sname})
    info=list(info1)
    return render_template('shome.html',data=info)
@app.route('/upstatus/<x>',methods=['POST','GET'])
def upstatus(x):
    name2=x
    status="Accepted"
    urequests.update_one({'_id':ObjectId(name2)},{'$set':{'status':status}})
    return redirect('/shome')
@app.route('/uhome',methods=['POST','GET'])
def uhome():
    global loc
    print(crop)
    if(loc=="vijayawada"):
        loc="krishna"
    info=agroDealers.find({'loc':loc.lower()})
```

```
    data1=list(info)
    if len(data1)>0:
        dealers=[]
        for i in data1:
            dummy=[]
            dummy.append(i.get('name'))
            dummy.append(i.get('loc'))
            dummy.append(i.get('lm'))
            dummy.append(i.get('phno'))
            dealers.append(dummy)
        return render_template('uhome.html',data=dealers)
    else:
        msg="No shops found under your Location"
        return render_template('uhome.html',msg=msg)
@app.route('/sdata')
def sdata():
    msg1=""
    global loc
    global gtemp
    global gli
    global hum
    # print(loc)
    if(loc=="krishna"):
        loc="vijayawada"
    url =
'https://api.openweathermap.org/data/2.5/weather?q={}&appid={}'.format(loc,'5cc01
12b7233c62b228f04428e2a2163')
    res = requests.get(url)
    data1 = res.json()
    temp = data1['main']['temp']
    temp1=int(temp-273.15)
    if temp1>=70:
        msg="It's too hot!!!"
```

```
    elif temp1<70:
        msg="It's fine"
    try:
        response = requests.get(url)
        data = response.json()
        visibility = data['visibility']  # Visibility in meters
        cloudiness = data['clouds']['all']  # Cloud cover percentage
        light_intensity = int(visibility * (1 - cloudiness / 100)/10)
    except Exception as e:
        light_intensity = None
        print(f"Error: {e}")
    API_KEY = '5cc0112b7233c62b228f04428e2a2163'
    url2 = f'http://api.openweathermap.org/geo/1.0/direct?q={loc}&limit=1&appid={API_KEY}'
    response1 = requests.get(url2)
    if response1.status_code == 200:
        data = response1.json()
        latitude = data[0]['lat']
        longitude = data[0]['lon']
        url1 = f'http://api.openweathermap.org/data/2.5/air_pollution?lat={latitude}&lon={longitude}&appid={API_KEY}'
        response = requests.get(url1)
        if response.status_code == 200:
            data = response.json()
            if 'list' in data:
                air_quality = data['list'][0]['main']['aqi']
                components=data['list'][0]['components']
                # print('Air Quality Index (AQI):', air_quality)
                # print('components:',data['list'][0]['components'])
            else:
                msg1='No air quality data available for the specified location.'
```

```
        else:
            msg1='Failed to fetch air quality data. Please check your API key and try again.'
    else:
        msg1='Failed to fetch city data. Please check your API key and try again.'
    gtemp=temp1
    gli=light_intensity
    hum=data1['main']['humidity']
    return
render_template('sensor.html',data=data1,temp=temp1,msg=msg,msg1=msg1,li=light
_intensity,aq=air_quality,com=components)
@app.route('/dis')
def dis():
    return render_template('disease.html')
@app.route('/disease', methods=['POST', 'GET'])
def disease():
    if request.method == 'POST':
        img = request.files['image']
        if img:
            img_path = "static/" + img.filename
    img.save(img_path)
    np.set_printoptions(suppress=True)
    model = load_model("keras_model.h5", compile=False)
    class_names = open("labels.txt", "r").readlines()
    data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)
    image = Image.open(img).convert("RGB")
    size = (224, 224)
    image = ImageOps.fit(image, size, Image.Resampling.LANCZOS)
    image_array = np.asarray(image)
    normalized_image_array = (image_array.astype(np.float32) / 127.5) - 1
    data[0] = normalized_image_array
    prediction = model.predict(data)
    index = np.argmax(prediction)
    class_name = class_names[index]
```

```
    confidence_score = prediction[0][index]
    result = f"Class: {class_name[2:]}, Confidence Score: {confidence_score}"
    # print(result)
    reasons_index = result.find("Reasons=[")
    na=result[2:reasons_index-1]
    # print("name:",na)
    remedies_index = result.find("Remedies=[")
    if reasons_index != -1:
        reasons_end_index = result.find("]", reasons_index)
        reasons_str = result[reasons_index + len("Reasons=["):reasons_end_index]
        reasons = [reason.strip() for reason in reasons_str.split(",")]
    if remedies_index != -1:
        remedies_end_index = result.find("]", remedies_index)
    remedies_str=result[remedies_index+len("Remedies=["):remedies_end_index]
        remedies = [remedy.strip() for remedy in remedies_str.split(",")]
    # print("Reasons:", reasons)
    # print("Remedies:", remedies)
            return
render_template('disease.html',na=na[5:],reasons=reasons,remedies=remedies,confide
nce_score=confidence_score,img=img_path)
        else:
            return render_template('disease.html', result="Image file not accepted")
    else:
        return render_template('disease.html', result=None)
@app.route('/toone',methods=['POST','GET'])
def toone():
    info=indiBuyers.find()
    data1=list(info)
    if len(data1)>0:
        dealers=[]
        for i in data1:
            dummy=[]
            dummy.append(i.get('name'))
```

```
            dummy.append(i.get('loc'))
            dummy.append(i.get('phno'))
            dealers.append(dummy)
        return render_template('indibuyers.html',data=dealers)
    else:
        msg="No Buyers found"
        return render_template('indibuyers.html',msg=msg)
@app.route('/sell1',methods=['POST','GET'])
def sell1():
    sname=request.form.get('item')
    cname1=request.form.get('cname')
    cquan1=request.form.get('cquan')
    type="indi"
edate=str(datetime.now().day)+"/"+str(datetime.now().month)+"/"+str(datetime.now()
.year)
    exis_req=urequests.find_one({'sname':sname,'name':name,'cname':cname1})
    if exis_req:
        if exis_req['status']=="pending":
            return redirect('/toone')
    else:
        status="pending"
new_req={'sname':sname,'name':name,'cname':cname1,'cquan':cquan1,'type':type,'dat
e':edate,'status':status}
        urequests.insert_one(new_req)
        msg="Request sent successfully!!!"
        print(msg)
        return redirect('/toone')
@app.route('/cprofile',methods=['POST','GET'])
def cprofile():
    global cname
    global cphno
    name1=request.form.get('newName')
    age1=request.form.get('newAge')
```

```
    loc1=request.form.get('newLoc')
    lm1=request.form.get('newLm')
    ph1=request.form.get('newPh')
    if name1!=None and name1!="":
        indiBuyers.update_one({'phno':cphno},{'$set':{'name':name1}})
        cname=name1
        print("details updated")
    elif age1!=None and age1!="" :
        indiBuyers.update_one({'phno':cphno},{'$set':{'age':age1}})
        print("details updated")
    elif loc1!=None and loc1!="" :
        indiBuyers.update_one({'phno':cphno},{'$set':{'loc':loc1}})
        print("details updated")
    elif lm1!=None and lm1!="":
        indiBuyers.update_one({'phno':cphno},{'$set':{'landm':lm1}})
        print("details updated")
    elif ph1!=None and ph1!="":
        indiBuyers.update_one({'name':cname},{'$set':{'phno':ph1}})
        cphno=ph1
        print("details updated")
    else:
        data=indiBuyers.find({'phno':cphno,'name':cname})
        data1=list(data)
    data=indiBuyers.find({'phno':cphno,'name':cname})
    data1=list(data)
    print(name1,age1,loc1,lm1,ph1)
    return render_template('cprofile.html',data=data1)
@app.route('/crequests',methods=['POST','GET'])
def crequest1():
    info1=urequests.find({'type':'indi','name':name})
    info=list(info1)
    return render_template('crequests.html',data=info)
```

```
@app.route('/cupstatus1/<y>',methods=['POST','GET'])
def cupstatus1(y):
    name2=y
    urequests.delete_one({'_id':ObjectId(name2)})
    return redirect('/crequests')
@app.route('/toemart',methods=['POST','GET'])
def toemart():
    info=emarts.find()
    data1=list(info)
    if len(data1)>0:
        dealers=[]
        for i in data1:
            dummy=[]
            dummy.append(i.get('path'))
            dealers.append(dummy)
        return render_template('sellingm.html',data=dealers)
    else:
        msg="No platforms available"
        return render_template('sellingm.html',msg=msg)
@app.route('/tobidding',methods=['POST','GET'])
def tobidder():
    info=biddings.find()
    data1=list(info)
    if len(data1)>0:
        dealers=[]
        for i in data1:
            dummy=[]
            dummy.append(i.get('path'))
            dealers.append(dummy)
        return render_template('sellingm.html',data=dealers)
    else:
        msg="No platforms available"
        return render_template('sellingm.html',msg=msg)
```

```
@app.route('/stubble',methods=['POST','GET'])
def stubble():
    info=bioIndus.find()
    data1=list(info)
    moksha=['Moksha Bio Energy Products','Vijayawada','+91 988 533 5864']
    indus=[]
    indus.append(moksha)
    for i in data1:
        dummy=[]
        dummy.append(i.get('name'))
        dummy.append(i.get('loc'))
        dummy.append(i.get('phno'))
        indus.append(dummy)
    return render_template('stubble.html',data=indus,msg=msg1)
@app.route('/stubble1',methods=['POST','GET'])
def stubble1():
    global sname
    sname=request.form.get('item')
    cname=request.form.get('cname')
    cquan=request.form.get('cquan')
    type="stubble"
edate=str(datetime.now().day)+"/"+str(datetime.now().month)+"/"+str(datetime.now()
.year)
    exis_req=urequests.find_one({'sname':sname,'name':name,'cname':cname})
    if exis_req:
        if exis_req['status']=="pending":
            return redirect('/stubble')
    else:
        status="pending"
new_req={'sname':sname,'name':name,'cname':cname,'cquan':cquan,'type':type,'date':e
date,'status':status}
        urequests.insert_one(new_req)
        msg="Request sent successfully!!!"
```

```
        return redirect('/stubble')
@app.route('/requests',methods=['POST','GET'])
def request1():
    info1=urequests.find({'type':'stubble','name':name})
    info=list(info1)
    return render_template('requests.html',data=info)
@app.route('/upstatus1/<y>',methods=['POST','GET'])
def upstatus1(y):
    name2=y
    urequests.delete_one({'_id':ObjectId(name2)})
    return redirect('/requests')
@app.route('/profile',methods=['POST','GET'])
def profile():
    global name
    global phno
    name1=request.form.get('newName')
    age1=request.form.get('newAge')
    loc1=request.form.get('newLoc')
    lm1=request.form.get('newLm')
    ph1=request.form.get('newPh')
    crop1=request.form.get('newCrop')
    if name1!=None and name1!="":
        userdata.update_one({'phno':phno},{'$set':{'name':name1}})
        name=name1
        print("details updated")
    elif age1!=None and age1!="" :
        userdata.update_one({'phno':phno},{'$set':{'age':age1}})
        print("details updated")
    elif loc1!=None and loc1!="" :
        userdata.update_one({'phno':phno},{'$set':{'loc':loc1}})
        print("details updated")
    elif lm1!=None and lm1!="":
        userdata.update_one({'phno':phno},{'$set':{'landm':lm1}})
```

```
    print("details updated")
elif crop1!=None and crop1!="":
    userdata.update_one({'phno':phno},{'$set':{'crop':crop1}})
    print("details updated")
elif ph1!=None and ph1!="":
    # print(name)
    userdata.update_one({'name':name},{'$set':{'phno':ph1}})
    phno=ph1
    print("details updated")
else:
    data=userdata.find({'phno':phno,'name':name})
    data1=list(data)
data=userdata.find({'phno':phno,'name':name})
data1=list(data)
print(name1,age1,loc1,lm1,ph1,crop1)
return render_template('profile.html',data=data1)
```

CHAPTER – 5: TESTING

5.1 Introduction to Testing

Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.

Software testing is an investigation conducted to provide stack holders with information about the quality of the software product or services under test. Software testing can also provide an objective independent view of the software to allow the business to appreciate and understand the risk of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects) and verifying that the software product is fit for use.

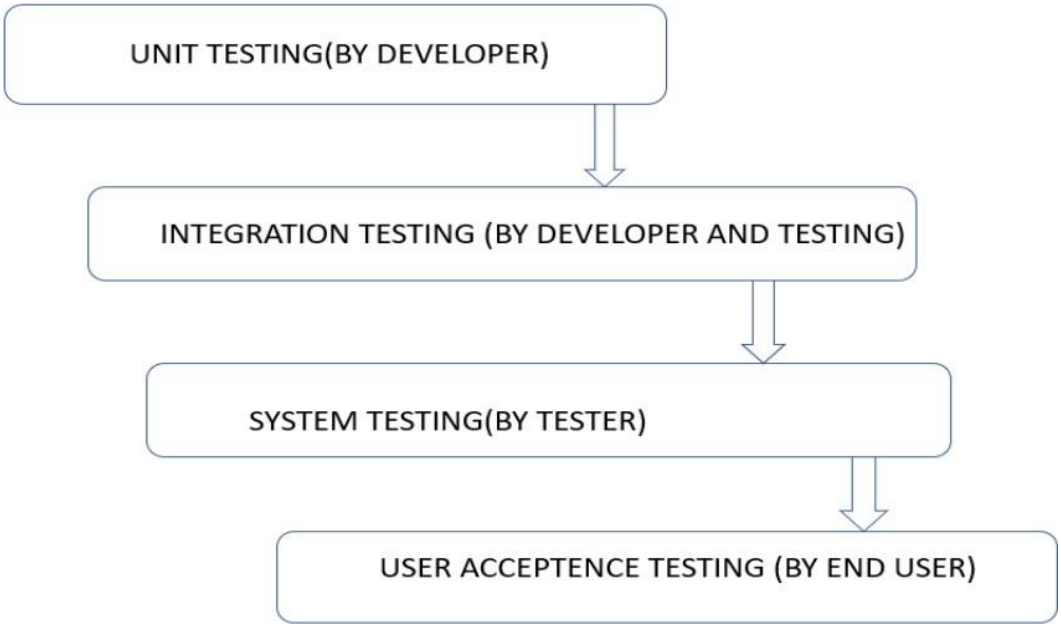
Software testing involves the execution of a software component or system component to evaluate one or more properties of interest, in general, these properties indicate the extent to which the component or system under test;

- To meet the requirements that guided its design and development.
- It responds correctly to all kinds of inputs.
- It performs its functions within an acceptable time.
    - It is sufficiently usable.
    - It can be installed and run in its intended environments and
    - To achieve the general, result its stockholder's desire.
    - As the number of possible tests for even simple software components is practically infinite, all software using uses some strategy to select tests that are feasible for the available time and resources. As a result, software testing typically (but not exclusively) attempts to execute a program or application with the intent of finding software bugs (errors or other defects). The job of testing is an iterative process as when one bug is fixed, it can illuminate other, deeper bug, or can even create new once Software testing can be conducted as soon as executable software (even if partially complete) exits.
    - Software testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is defect free.
    - It involves execution of a software component or system component or system component to evaluate one or more properties of interest.
    - Software testing also helps to identify errors, gaps or missing requirements in

contrary in the natural requirements.

- It can be either done manually or using automated tools.
- Some prefer saying software testing as a white box and black box testing.
- In simple terms, software testing means verification of application under test (AUT).
- Meets the software and technical requirements that guided its design and Development.
- Works as expected.
- Can be implemented with the same characteristics.

Fig-5.1: Testing levels



5.2 Types of Testing

Unit testing:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfied, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

System testing:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Acceptance Testing:

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements. Module Testing: Module testing will be done to the test the interaction between the various programs within the module. It checks the functionality of each program with relation to other programs within to other programs within the same module. It then tests the functionality of the module.

5.3 Test cases and Test Reports

Test Case: A TEST CASE is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly. The process of developing test cases can also help find problems in the requirements or design of an application.

How to write test cases for software:

- Use a Strong Title.
- Include a Strong Description.
- Include Assumptions and Preconditions.
- Keep the Test Steps Clear and Concise.
- Include the Expected result.
- Make it Reusable.

5.3.1 Farmer Sign up

Test Case id	Testing Unit	Test Data	Expected Output	Test Results	Action	Description
Tid 1	Enter all the details of the user	Username: gowri Password: ********** (all required)	Successfully Signed up	Pass	Storing details in database and creating a new id	Creating a new user data

Table-5.1: Test case for farmer sign-up

5.3.2 Test cases for Farmer Login are as follows:

Test Case id	Testing Unit	Test Data	Expected Output	Test Results	Action	Description
Tid 2	Enter correct Username & correct password	Username: gowri Password: **********	Login Successfully	Pass	Opens the user details	Check whether the credentials are correct or not
Tid 3	Enter correct Username & Incorrect Password	Username: gowri Password: *********	valid username & invalid password	Fail	It does not open the user details	Check whether the credentials are correct or not
Tid 4	Enter Incorrect Username & Correct Password	Username: gowr Password: **********	Invalid username & valid password	Fail	It does not open the user details	Check whether the credentials are correct or not
Tid 5	Enter Incorrect Username & Incorrect Password	Username: gowr Password: ********	Invalid username & password	Fail	It does not user the login details	Check whether the credentials are correct or not

Table-5.2: Test case for farmer login

5.3.3 Sign up for the Individual Consumer

Test Case id	Testing Unit	Test Data	Expected Output	Test Results	Action	Description
Tid 6	Enter all the details of the user	Username: gowri Password: ********** (all required)	Successfully Signed up	Pass	Storing details in database and creating a new id	Creating a new user data

Table-5.3: Test case for consumer sign-up

5.3.4 Test cases for Individual Consumer Login are as follows:

Test Case id	Testing Unit	Test Data	Expected Output	Test Results	Action	Description
Tid 7	Enter correct Username & correct password	Username: gowri Password: **********	Login Successfully	Pass	Opens the user details	Check whether the credentials are correct or not
Tid 8	Enter correct Username & Incorrect Password	Username: gowri Password: *********	valid username & invalid password	Fail	It does not open the user details	Check whether the credentials are correct or not
Tid 9	Enter Incorrect Username & Correct Password	Username: gowr Password: **********	Invalid username & valid password	Fail	It does not open the user details	Check whether the credentials are correct or not
Tid 10	Enter Incorrect Username & Incorrect Password	Username: gowr Password: ********	Invalid username & password	Fail	It does not user the login details	Check whether the credentials are correct or not

Table-5.4: Test case for consumer login

5.3.5 Test cases for Bio-Industry Login are as follows:

Test Case Id	Testing Unit	Test Data	Expected Output	Test Results	Action	Description
Tid 11	Select correct Username	Username: Gowri industry	Login Successfully	Pass	Opens the user details	Check whether the credentials are correct or not
Tid 12	Select correct Username	Username: gowri indus	invalid username	Fail	It does not open the user details	Check whether the credentials are correct or not

Table-5.5: Test case for bio-industry login

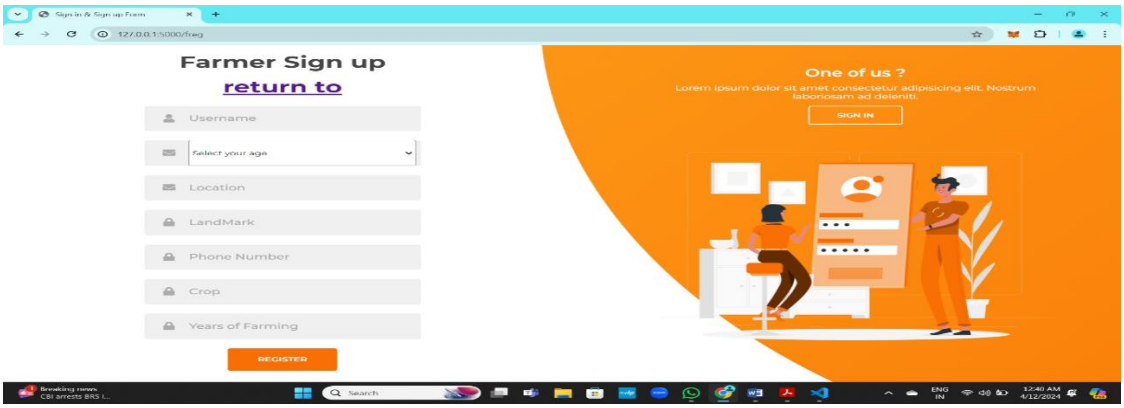
Test Reports:

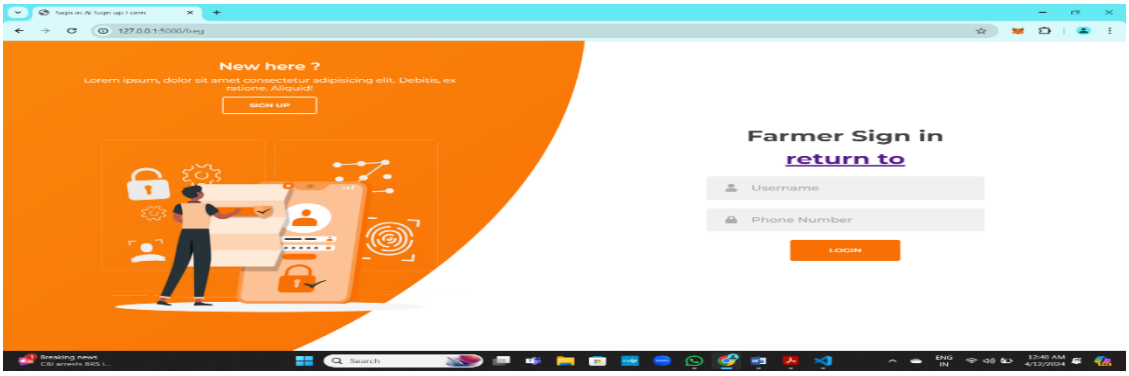
Purpose of test report:

Document that records data obtained from an experiment of Evaluation in an organized manner, describes the environmental or operating conditions, and shows the comparison of test results with test objectives.

Who prepares test summary report? Test summary report is a document which. contains Summary of test activities and final test results. After the testing cycle it is very important that you communicate the test results and findings to the project. stakeholders so that decisions can be made for the software release.

5.3.1 Test Report for Farmer Sign up are as follows:

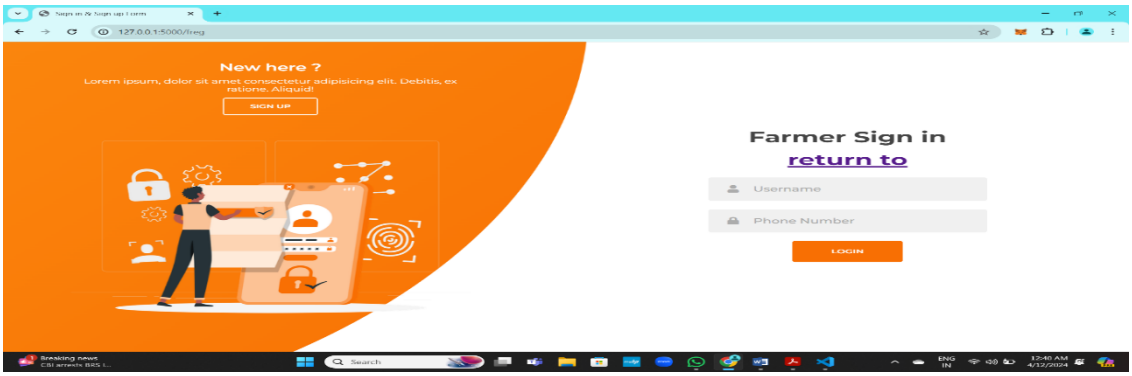


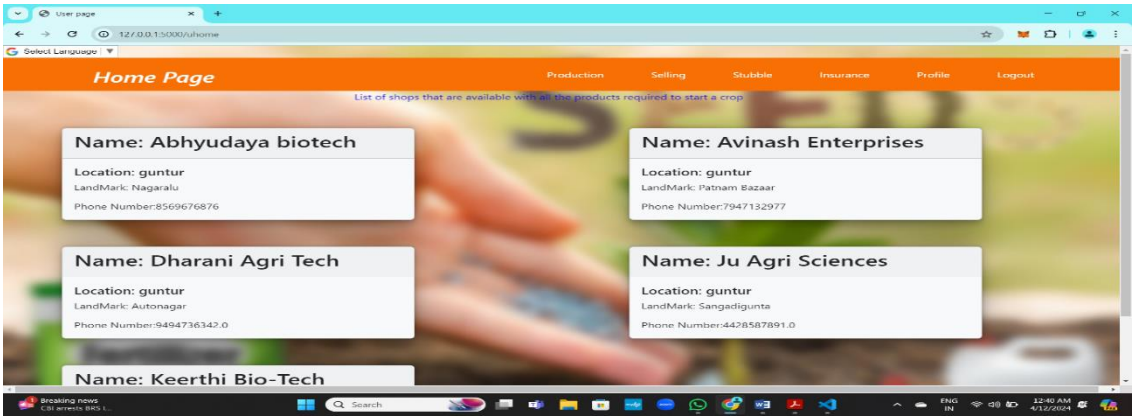


Test case no 1: To check whether the user has already had a sign up account or not.	**Priority:** High
Test objective: To check functionality and working of the Password constraints.	
Test Description: The main intention is providing valid length of password.	
Test Requirements Verified: Yes	
Test Environment: We will get the message displayed as “invalid details”.	
Test setup: Provide valid details.	
Actions: Click on sign-up button after providing valid details.	**Expected Result**: Redirection to the sign in page after providing valid credentials.
Problems: No	**Note**: Successfully Executed

Table-5.6: Test report for Farmer Sign up

5.3.2 Test Report for Farmer login are as follows:

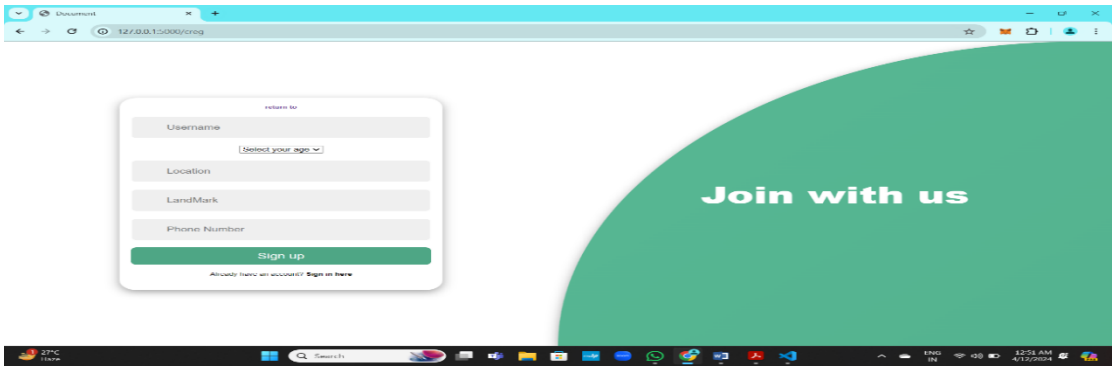


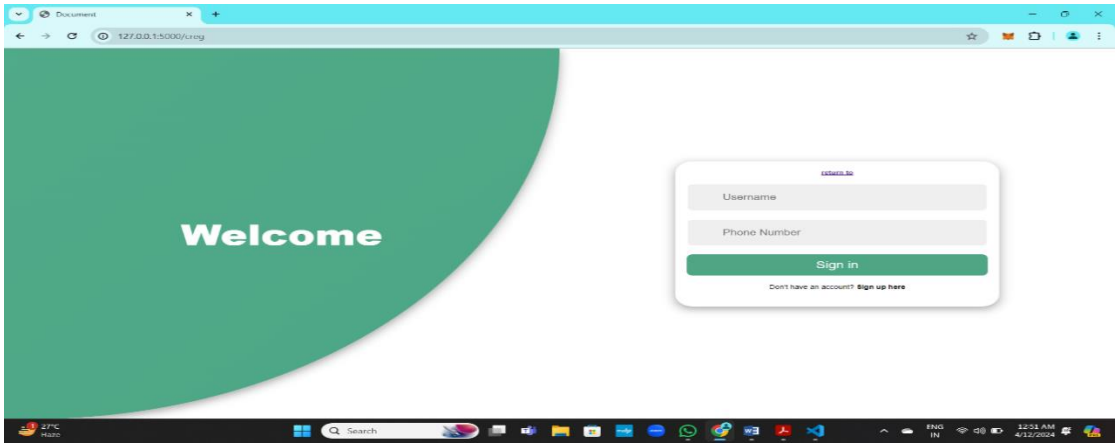


Test case no 2: To check whether the person logging into the application is an existing user	**Priority:** High
Test objective: To check functionality and working of login phase	
Test Description: The main intention of this to verify the credentials of the user	
Test Requirements Verified: Yes	
Test Environment: We will get data displayed in the screen as “invalid credentials”	
Test setup: Provide login id and password used during the registration	
Actions: Click on login button after providing credential	**Expected Result**: Redirection to a user login page.
Problems: No	**Note**: Successfully Executed

Table-5.7: Test report for Farmer login

5.3.3 Test Report for Individual Consumer Sign up are as follows:

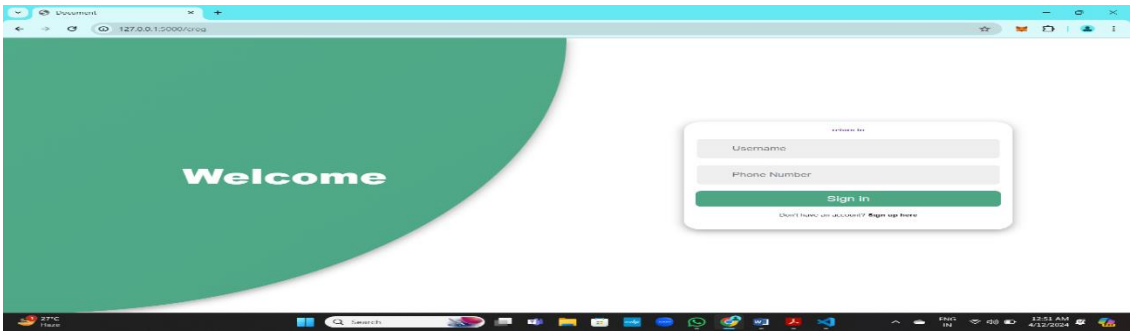


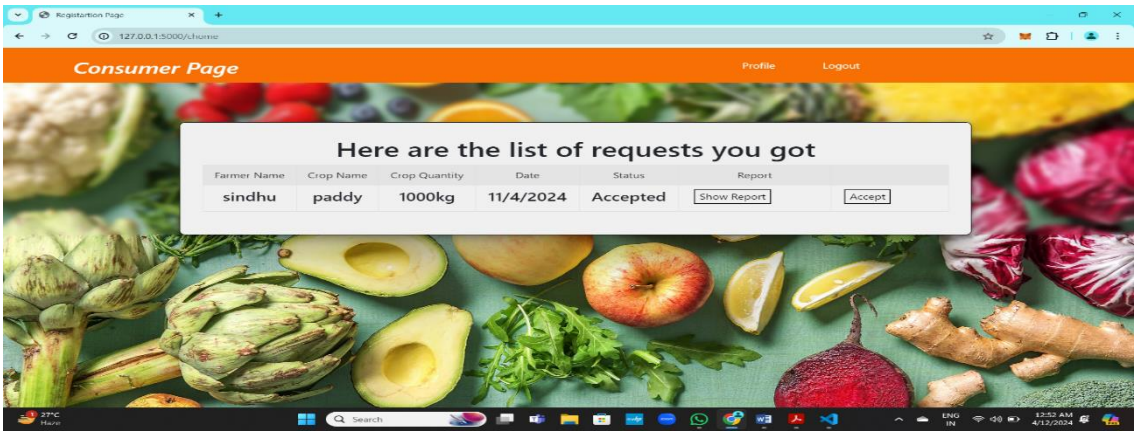


Test case no 1: To check whether the user has already had a sign up account or not.	**Priority:** High
Test objective: To check functionality and working of the Password constraints.	
Test Description: The main intention is providing valid length of password.	
Test Requirements Verified: Yes	
Test Environment: We will get the message displayed as “invalid details”.	
Test setup: Provide valid details.	
Actions: Click on sign-up button after providing valid details.	**Expected Result**: Redirection to the sign in page after providing valid credentials.
Problems: No	**Note**: Successfully Executed

Table-5.8: Test report for Consumer Sign up

5.3.4 Test Report for Individual Consumer login are as follows:

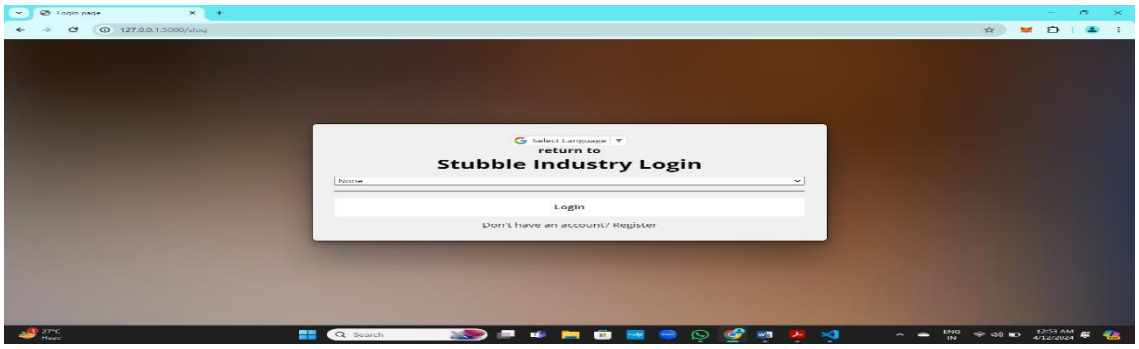


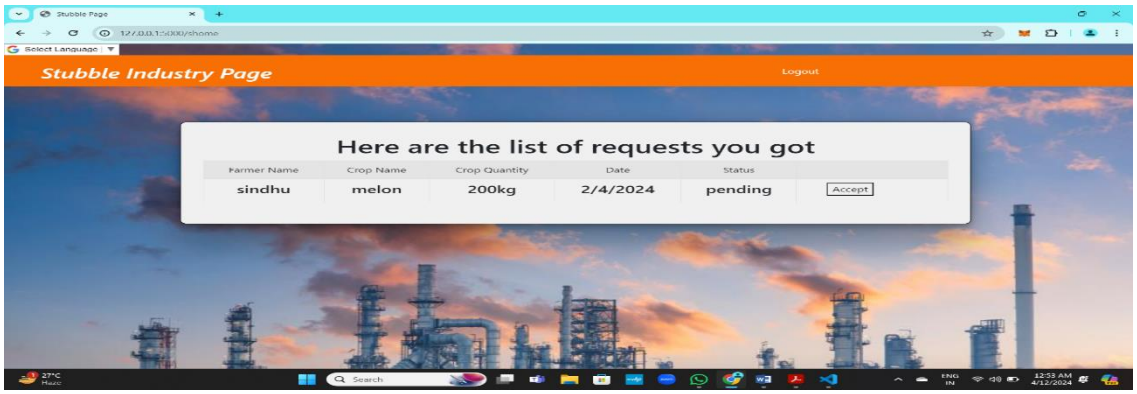


Test case no 3: To check whether the person logging into the application is an existing user	**Priority:** High
Test objective: To check functionality and working of login phase	
Test Description: The main intention of this to verify the credentials of the user	
Test Requirements Verified: Yes	
Test Environment: We will get data displayed in the screen as “invalid credentials”	
Test setup: Provide login id and password used during the registration	
Actions: Click on login button after providing credential	**Expected Result**: Redirection to a user login page.
Problems: No	**Note**: Successfully Executed

Table-5.9: Test report for Consumer login

5.3.5 Test Report for Bio-Industry login are as follows:





Test case no 4: To check whether the person logging into the application is an existing user	**Priority:** High
Test objective: To check functionality and working of login phase	
Test Description: The main intention of this to verify the credentials of the user	
Test Requirements Verified: Yes	
Test Environment: We will get data displayed in the screen as “invalid credentials”	
Test setup: Provide login id and password used during the registration	
Actions: Click on login button after providing credential	**Expected Result**: Redirection to a user login page.
Problems: No	**Note**: Successfully Executed

Table-5.10: Test report for bio-industry login

CHAPTER – 6: IMPLEMENTATION

6.1 Implementation Introduction

Implementation is the process of putting a plan, design, or idea into action. It involves taking theoretical concepts and making them a reality by executing a plan and bringing a product or service to life. In software development, implementation refers to the stage where the design and planning are translated into actual code and deployed into a production environment.

Implementation is a critical phase in any project as it determines the success or failure of the project. A well-planned and executed implementation ensures that the product or service meets the intended requirements, and is functional, reliable, and user-friendly. On the other hand, a poorly executed implementation can result in technical problems, delays, cost overruns, and damage to the reputation of the company. Therefore, it is essential to ensure that the implementation process is carefully planned and executed to minimize risks and ensure the successful completion of the project.

6.2 Implementation Procedure & Steps

Implementation is the carrying out, execution, or practice of a plan, a method, or any design, idea, model, specification, standard or policy for doing something. As such, implementation is the action that must follow any preliminary thinking in order for something to actually happen. Many preparations are involved before and during the implementation of the proposed system.

6.2.1 Implementation Steps

- Requirement Analysis.
- Hardware Setup
- Software Development
- Blockchain Integration
- Automation and Control Setup
- Crop Selling Mechanisms
- Stubble Utilization Integration
- Insurance Scheme Management
- Testing and Quality Assurance
- Deployment and Training
- Monitoring and Maintenance

6.2.2 Implementation Procedure

Project Planning:

- Define project objectives, scope, and requirements, considering the integration of blockchain, machine learning (ML), Internet of Things (IoT), and frontend and backend technologies.
- Create a project plan detailing timelines, milestones, and resource allocations for each technology component.

System Design:

- Design the system architecture, incorporating components for data collection, processing, storage, and presentation.
- Define the data schema and structures for MongoDB, ensuring compatibility with the system's requirements.

Development:

Frontend Development:

- Develop the user interface using HTML, CSS, JavaScript, and Bootstrap to create responsive and visually appealing web pages.
- Implement interactive features and user controls for data input, visualization, and interaction.

Backend Development:

- Use Python with frameworks like Flask or Django to build the backend server application.
- Implement APIs and endpoints for handling data requests, authentication, and business logic.

Blockchain Integration:

- Choose a suitable blockchain platform (e.g., Ethereum) and develop smart contracts for implementing crop selling mechanisms and stubble utilization.
- Integrate blockchain functionalities into the backend server application to record transactions and ensure transparency.

Machine Learning:

- Develop ML models using libraries like TensorFlow or scikit-learn for disease identification and crop management.
- Train the models using historical data and implement them within the backend server application for real-time analysis.

IoT Integration:

- Configure IoT sensors to collect data on soil parameters and atmospheric conditions.
- Develop scripts or applications to process and transmit sensor data to the backend server application for further analysis.

Testing:

- Conduct thorough testing of each component, including frontend UI/UX, backend functionality, blockchain integration, ML algorithms, and IoT data collection.
- Perform unit testing, integration testing, and end-to-end testing to identify and resolve any issues or bugs.

Deployment:

- Deploy the system on a cloud platform or dedicated server infrastructure, ensuring scalability, reliability, and security.
- Configure MongoDB databases and ensure data backup and recovery mechanisms are in place.
- Set up continuous integration/continuous deployment (CI/CD) pipelines for automated deployment and updates.

Monitoring and Optimization:

- Monitor system performance, resource utilization, and user feedback to identify areas for optimization.
- Optimize code, database queries, and system configurations to improve efficiency and responsiveness.

Documentation and Training:

- Document system architecture, design decisions, and implementation details for future reference and maintenance.
- Provide training to users and administrators on system usage, features, and best practices.

Maintenance and Support:

- Establish protocols for ongoing maintenance, updates, and support to address any issues or feature requests.
- Regularly review and update the system to incorporate new technologies, address security vulnerabilities, and enhance functionality.

6.3 User Manual

Welcome to the Enhanced Agricultural Management System! This user manual will guide you through the functionalities and features of the system, helping you optimize your agricultural practices effectively.

Logging In:

- Visit the system's login page and enter your credentials (ID and password) to access the dashboard.
- If you are a new user, click on the "Sign Up" option to create a new account by providing your name, email ID, and password.

Dashboard:

- Upon logging in, you will be directed to the dashboard, where you can view key information and access various features of the system.

Data Collection and Monitoring:

- Navigate to the "Data Collection" section to monitor real-time data on soil parameters (fertility, moisture) and atmospheric conditions (air quality, temperature, light intensity) collected by IoT sensors.
- Use the interactive charts and graphs to analyze crop conditions and make informed decisions.

Automation Irrigation Control:

- In the "Irrigation Control" section, enable automation based on moisture data collected by sensors to optimize irrigation practices.
- Adjust irrigation settings as needed to ensure optimal water usage and crop health.

Disease Identification and Remediation:

- Explore the "Disease Identification" feature powered by machine learning algorithms to detect crop diseases.
- Receive organic remedy recommendations to manage and mitigate crop diseases effectively.

Crop Selling Mechanisms:

- Access the "Crop Selling" section to explore three methods for selling crops: individual buyer transactions, smart contract-based bidding, and a digital market interface.
- Choose the selling method that best suits your needs and preferences for transparent and efficient transactions.

Stubble Utilization:

- Explore the "Stubble Utilization" feature to sell crop residues (stubble) to bioenergy-producing industries.
- Ensure transparent stubble sales transactions and securely store related data in the blockchain.

Insurance Scheme Management:

- Visit the "Insurance Schemes" section to explore various insurance schemes tailored to your needs.
- Securely store final insurance scheme reports in the blockchain for transparency and data integrity.

Profile Settings:

- Update your profile settings, including personal information, preferences, and notification settings, in the "Profile" section.

Logging Out:

- Click on the "Logout" button to securely log out of the system and end your session.

The detailed roadmap to use our project is given below**.**

For user-1(Farmer):

Step-1: Initially you will be at the home page of our project when you select farmer option you will land in below page. You can directly login if you already has an account or else you can register by clicking sign up button.

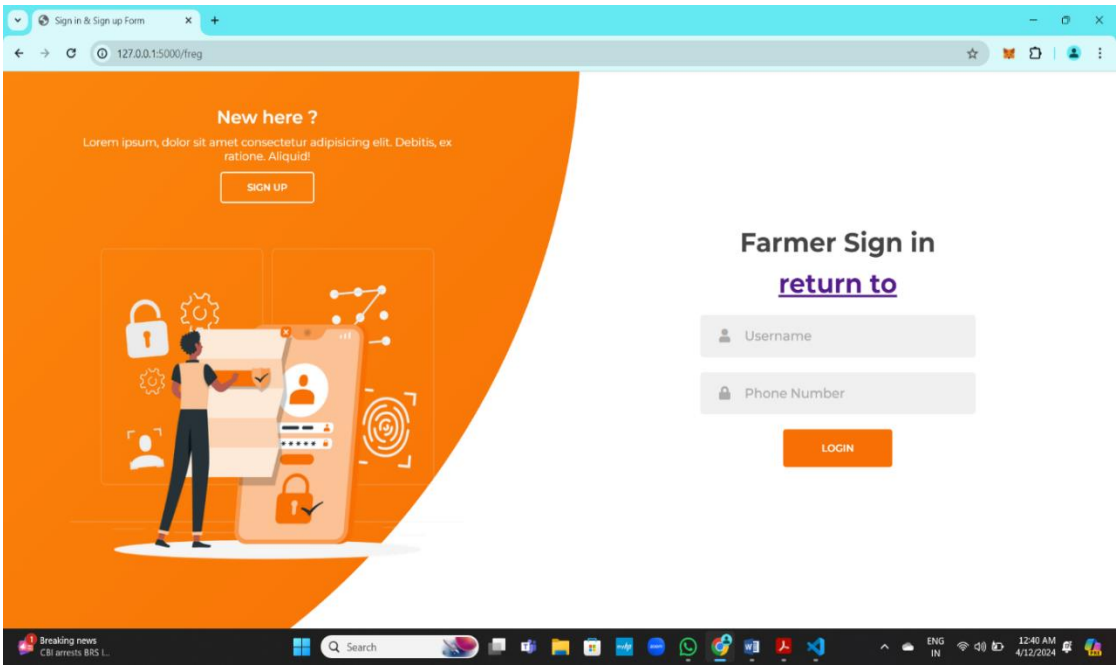


Fig-6.1: farmer login page

Step-2: After successful login you will land in main home page of the user as farmer there you have no.of options as per your requirement you can select one.

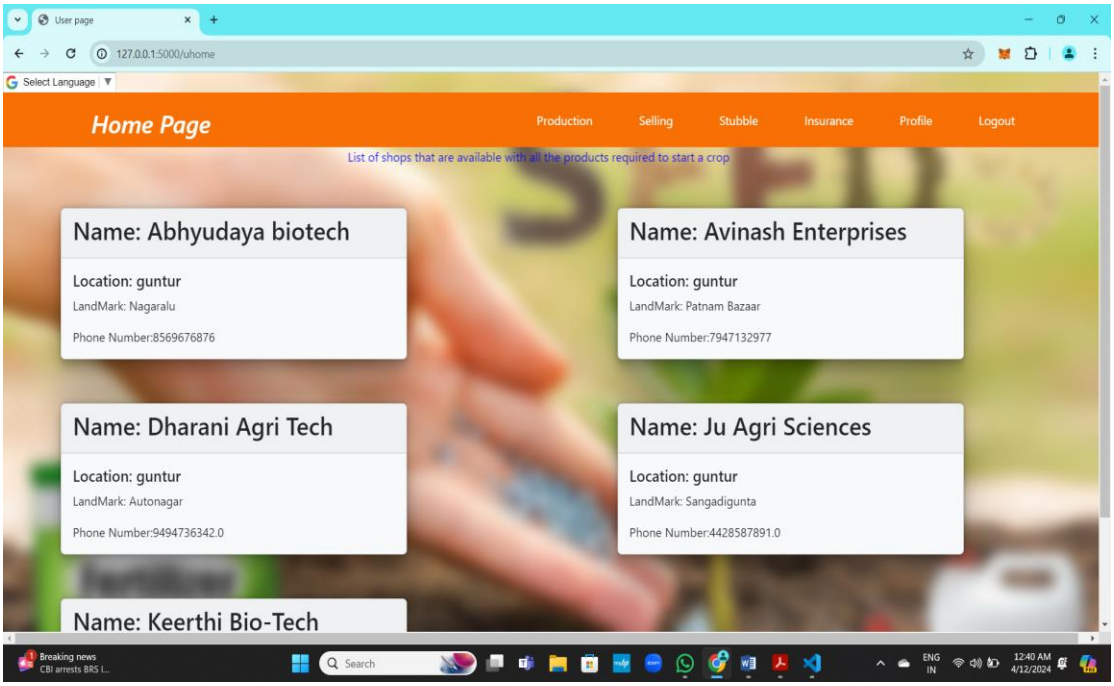


Fig-6.2: farmer home page

Step-3: If you select production then you will be navigated to below screen there you have 3 main options deals with sensor data, disease detection and irrigation system.

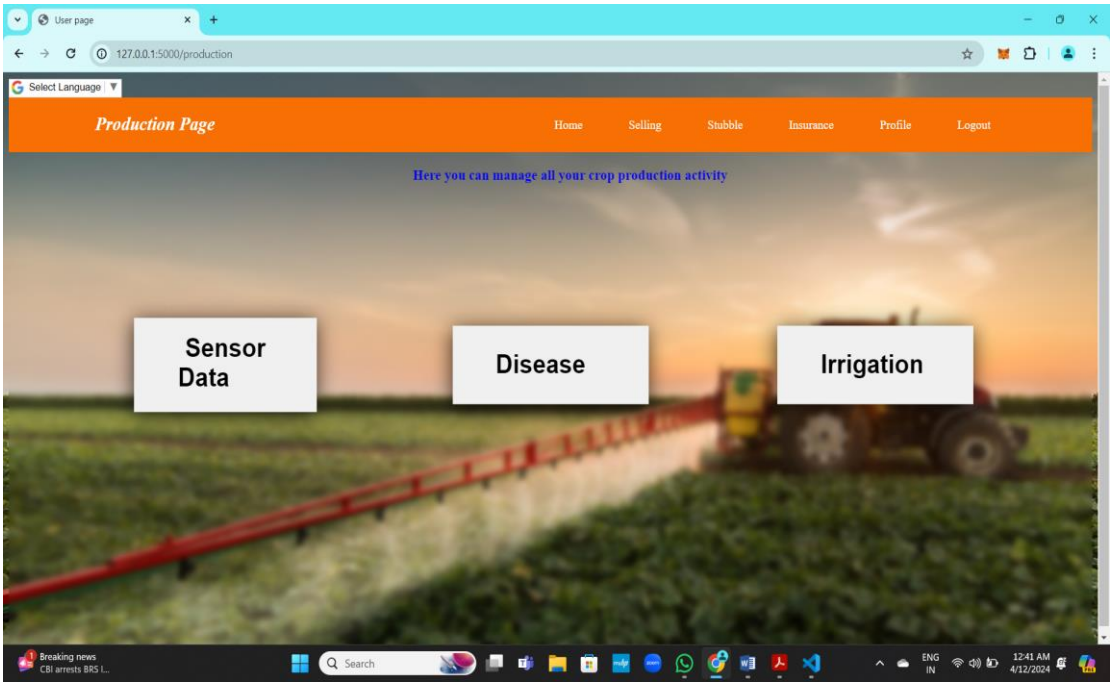


Fig-6.3: production page

Step-4: Now if you select selling option then you will land in below page where you can manage your crop selling activity where we have given 3 main options.

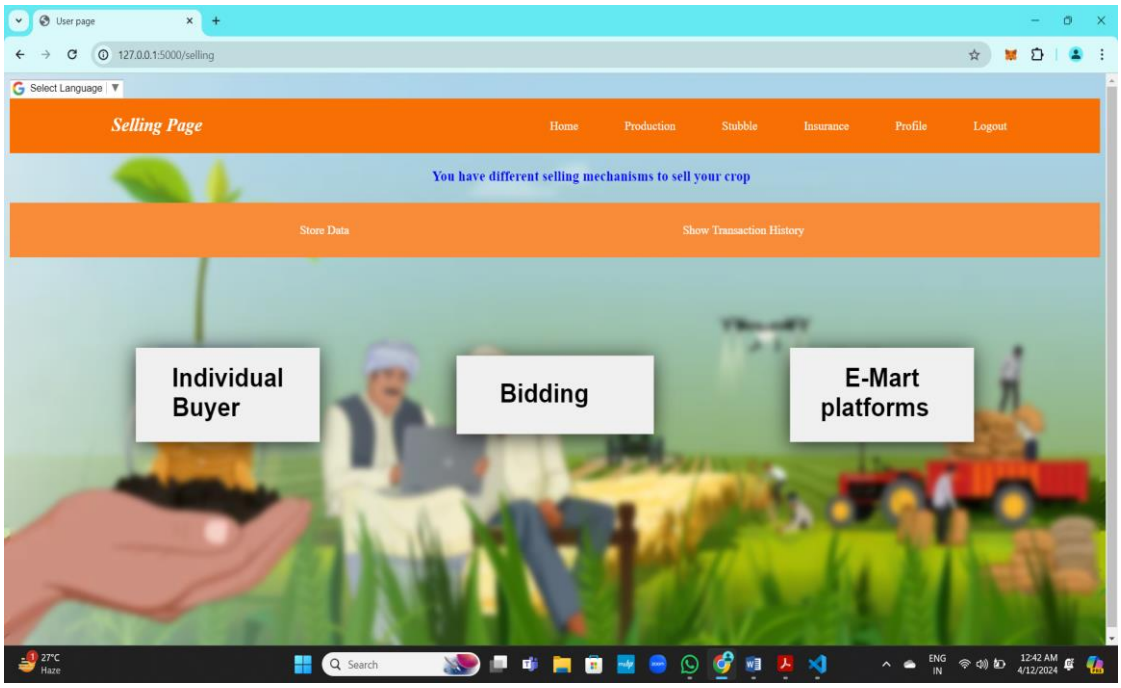


Fig-6.4: selling page

Step-5: If you select Stubble option then you can see the below page where you can send request to any bio-industry to sell your stubble to make additional income.

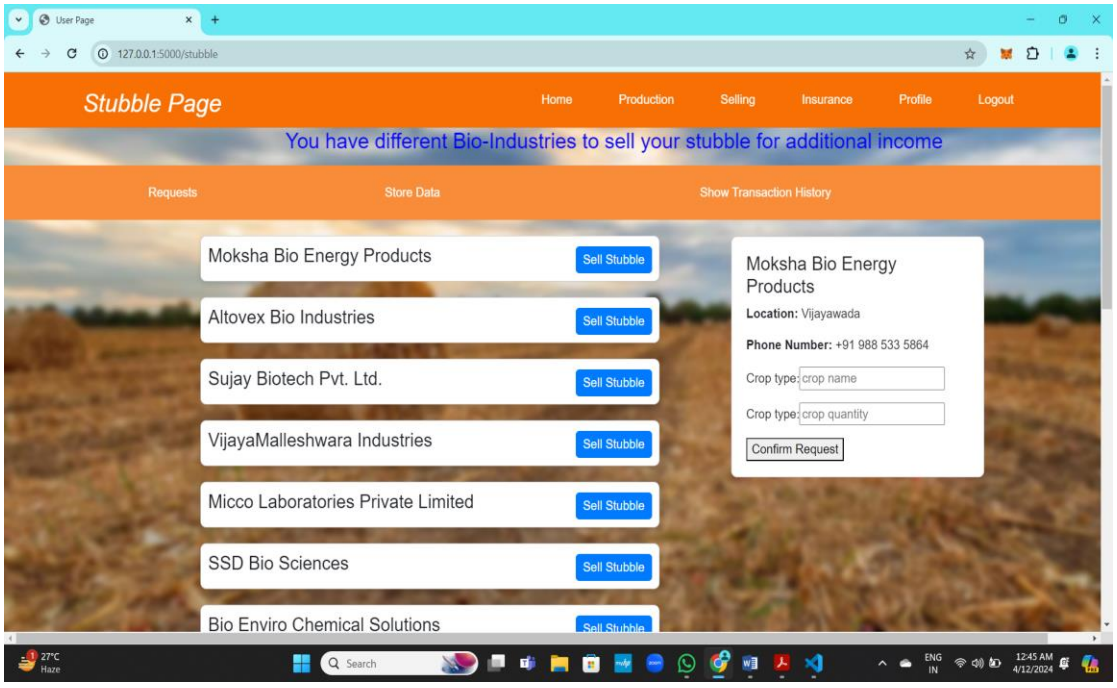


Fig-6.5: stubble page

Step-6: You are able to see list of different insurance schemes available by clicking on insurance which will be like this.

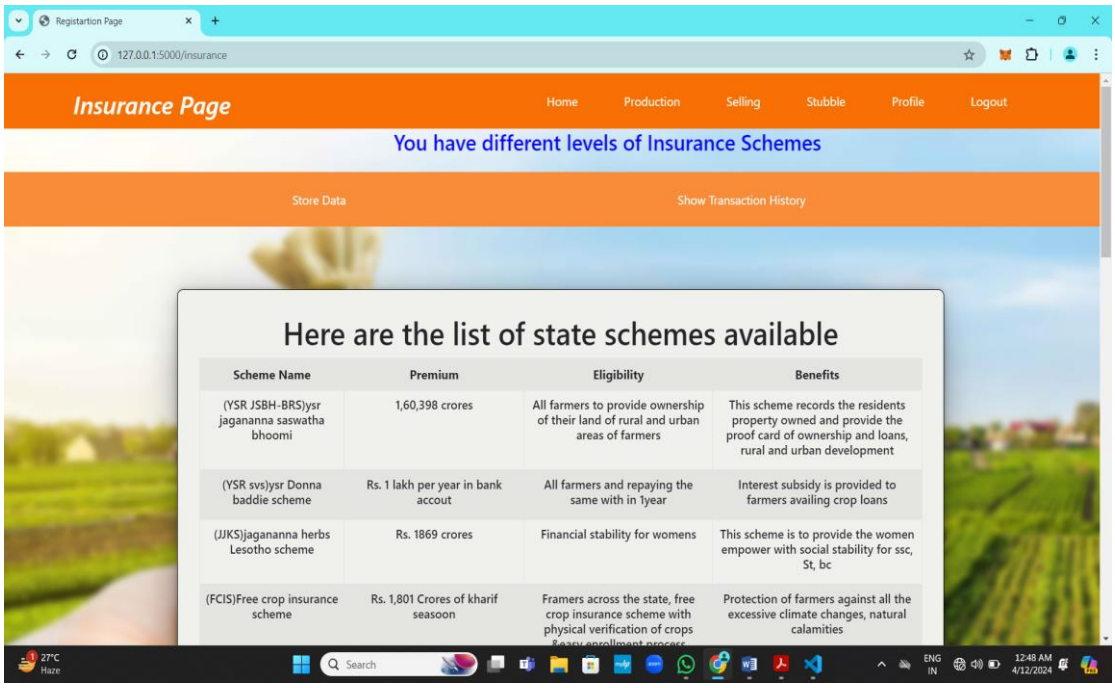


Fig-6.6: Insurance page

Step-7: You can view your profile details that are given while registering to the website by clicking on profile you can also change details.

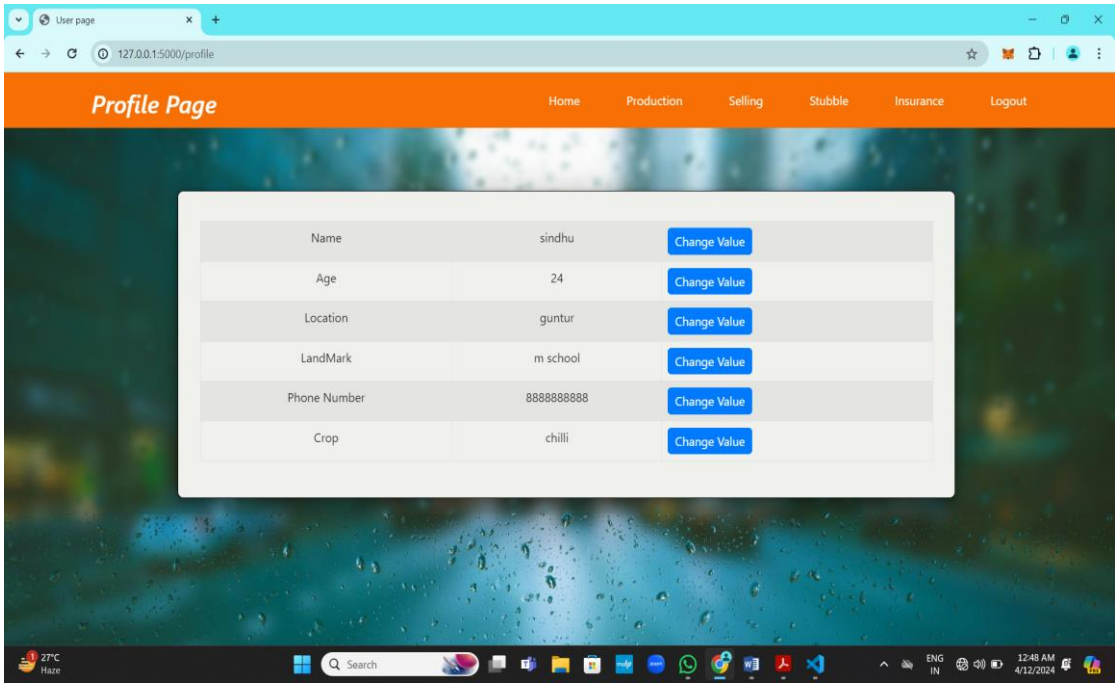


Fig-6.7: farmer profile page

Step-8: You can now logout by clicking on logout and can be returned to main home page of the website before you have logged in.

For user-2(Individual Consumer):

Step-1: Initially you will be at the home page of our project when you select Consumer option you will land in below page. You can directly login if you already has an account or else you can register by clicking sign up button.

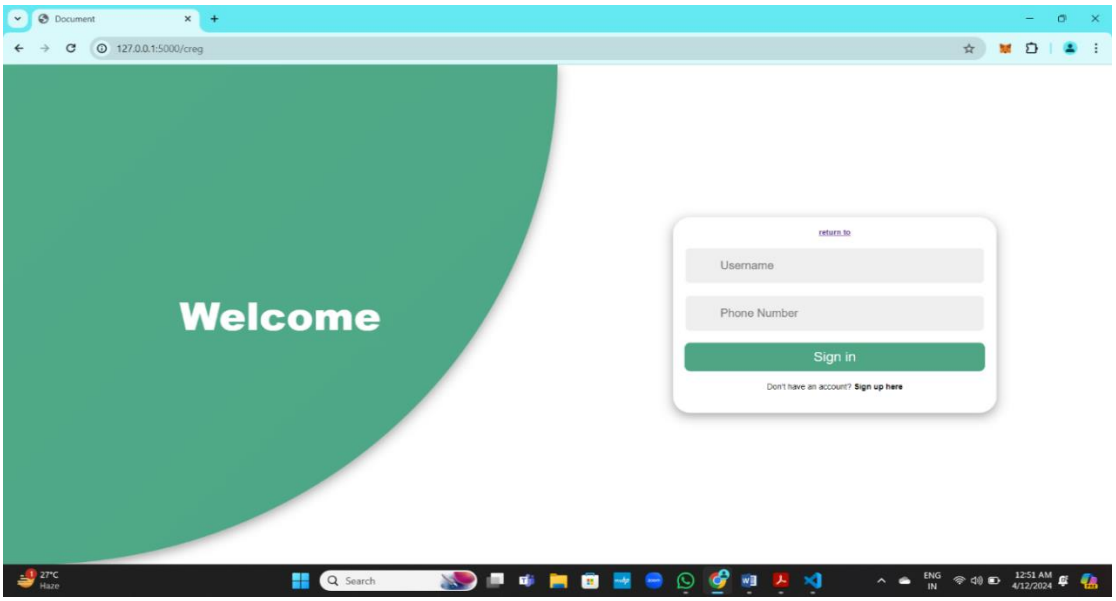


Fig-6.8: Individual login page

Step-2: After successful login you will land in main home page of the user as Consumer there you have requests that you got from the farmers and you can also see the report.

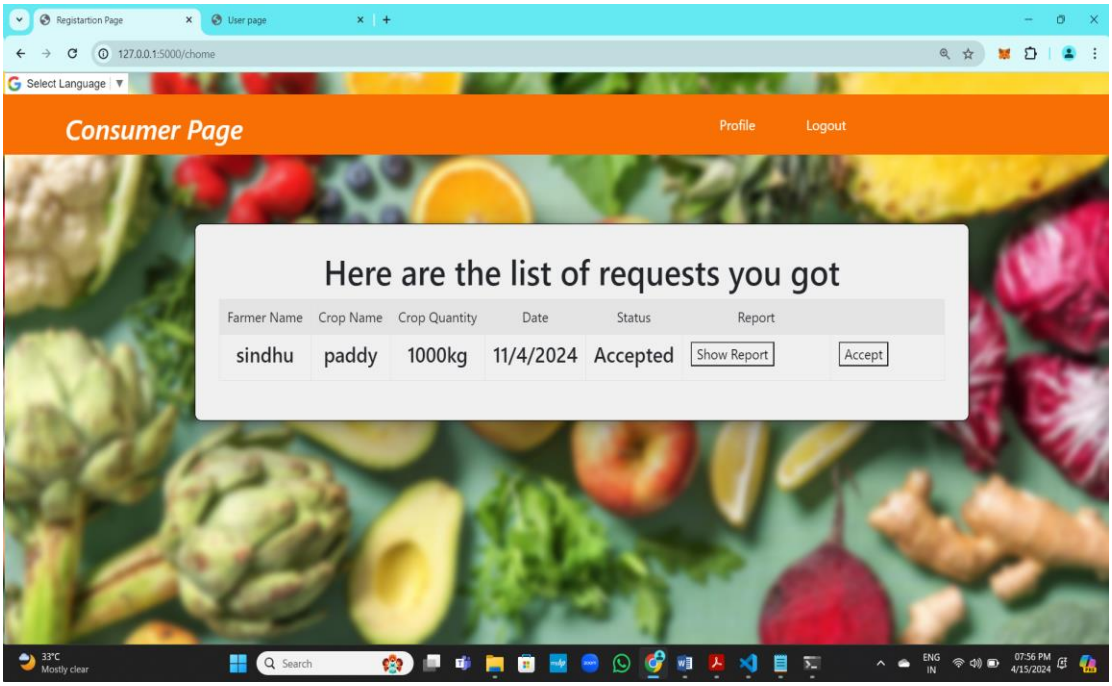


Fig-6.9: Consumer home page

Step-3: You can view the report of the particular farmer from whom you have got the request by clicking report button.

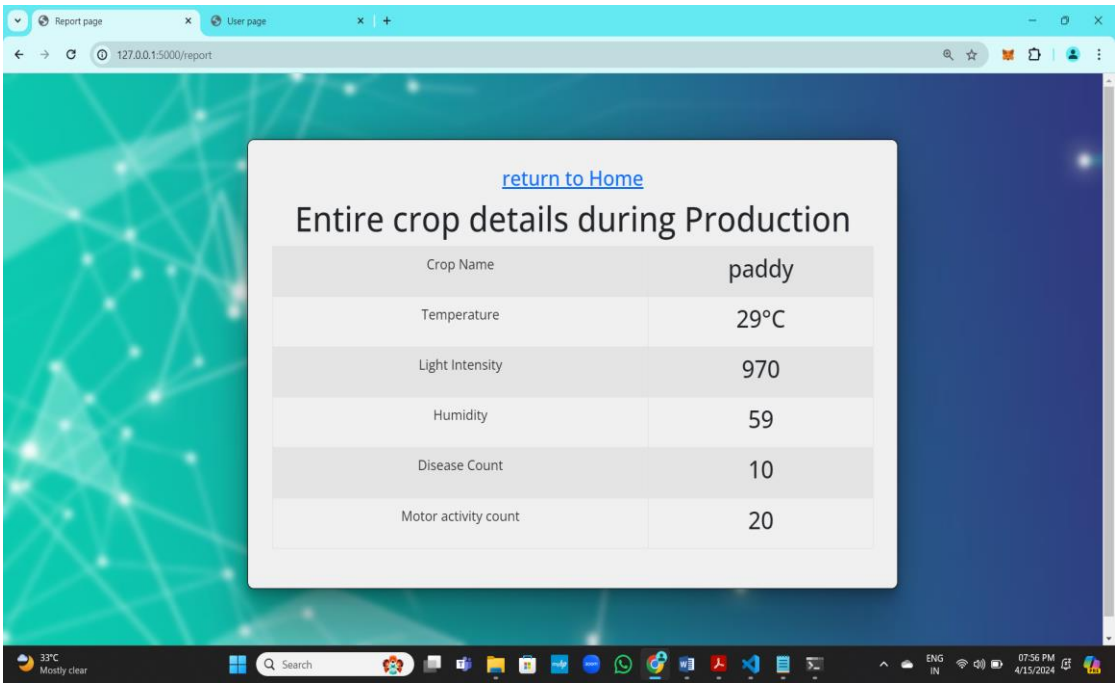


Fig-6.10: Report page

Step-4: You can view your profile details that are given while registering to the website by clicking on profile you can also change details.

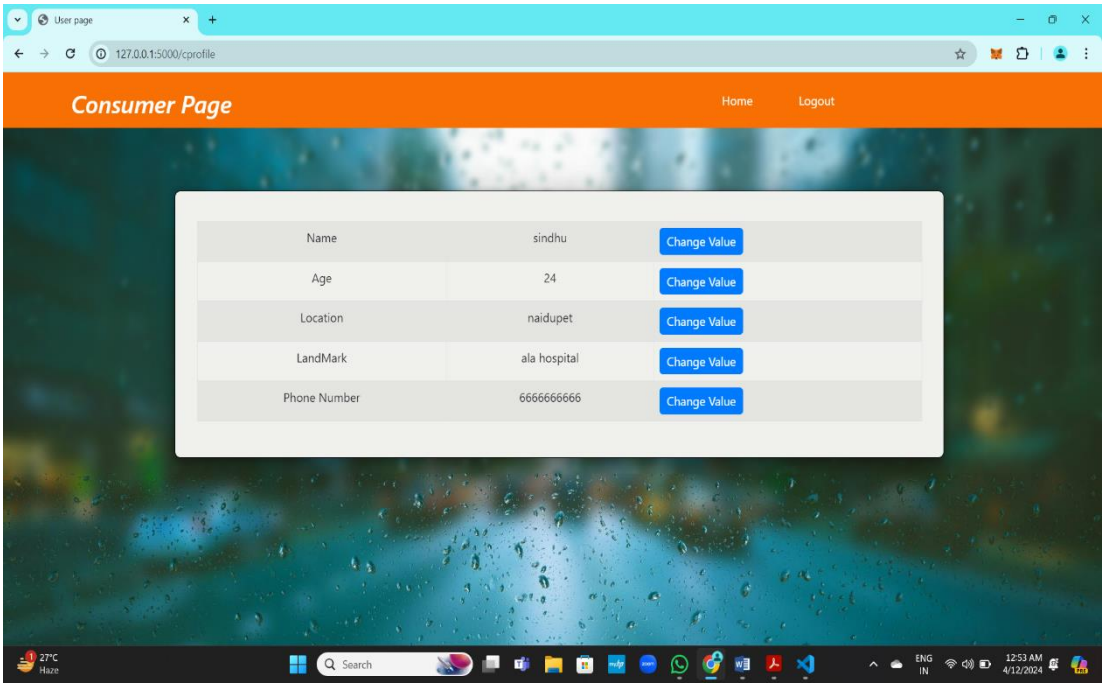


Fig-6.11: Consumer profile page

Step-5: You can now logout by clicking on logout and can be returned to main home page of the website before you have logged in.

For user-3(Bio-Industry):

Step-1: Initially you will be at the home page of our project when you select Bio-Industry option you will land in below page. You can directly login.

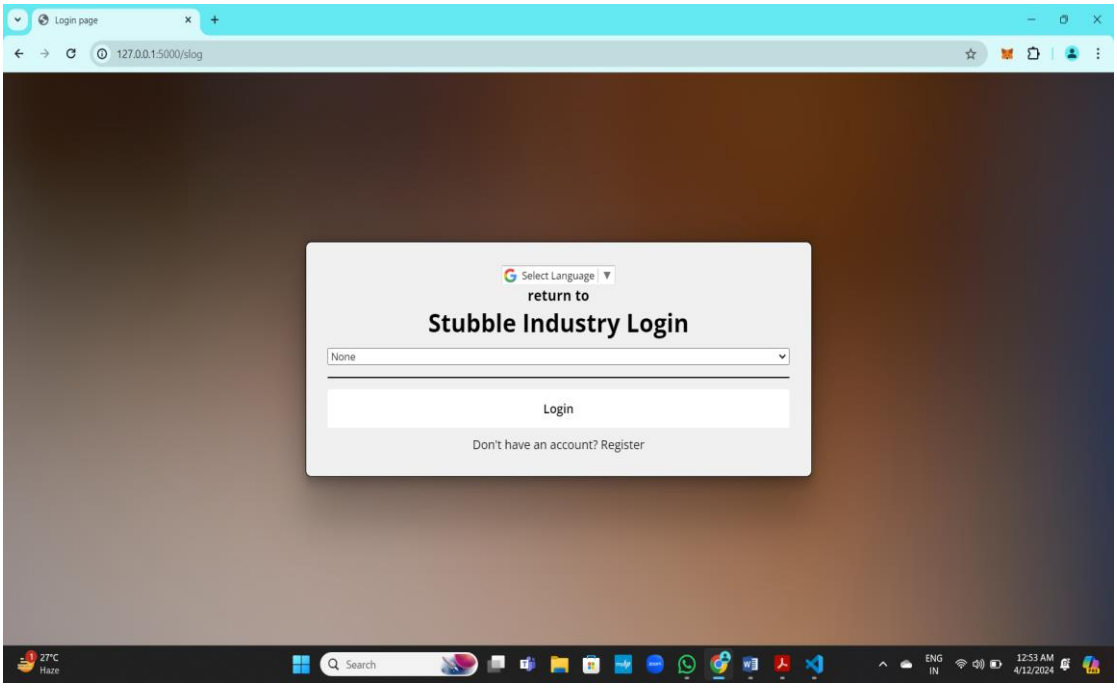


Fig-6.12: Bio-Industry login page

Step-2: You can view the list of requests you have got from the user and you can accept also.

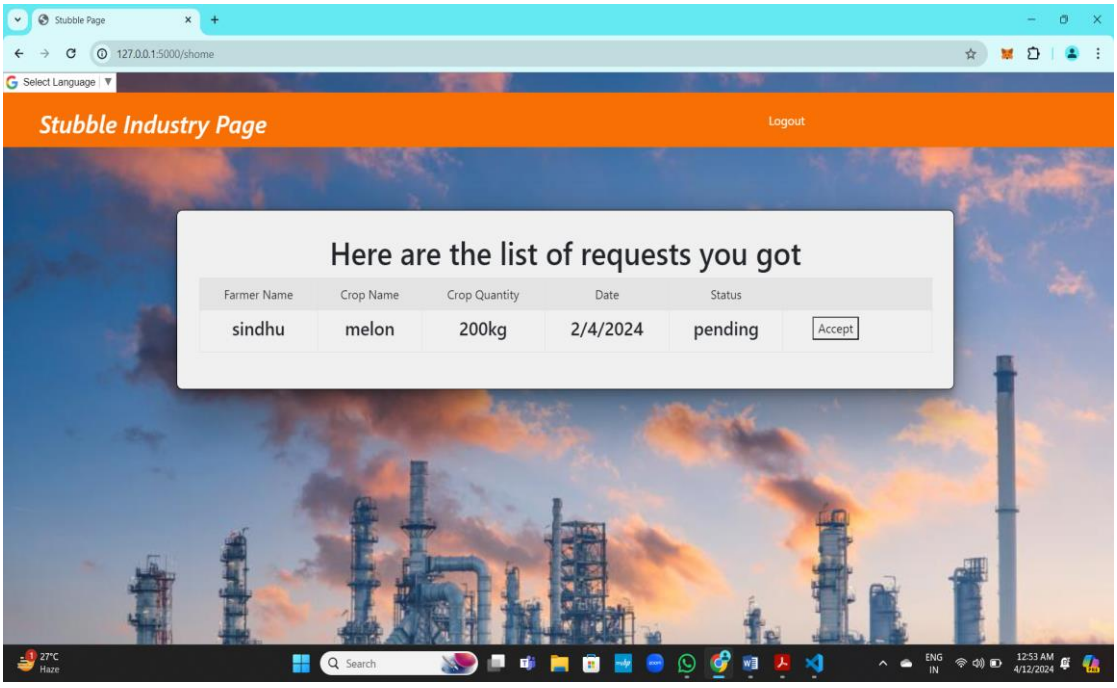


Fig-6.13: Bio-Industry home page

CHAPTER – 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

In summary, our agricultural project stands as a testament to the transformative potential of technology in addressing key challenges within the farming industry. By prioritizing user needs and leveraging advanced tools, we have developed a comprehensive solution that addresses critical aspects such as crop monitoring, irrigation management, selling mechanisms, and financial protection. Through collaboration with stakeholders and adherence to best practices, our project not only streamlines existing processes but also sets new standards for efficiency, transparency, and sustainability in agriculture. As we move forward, our commitment to innovation and continual improvement remains unwavering, driving us towards a future where farming is not just a livelihood but a thriving, resilient ecosystem supported by technology.

7.2 Future Enhancements

In the future, the agricultural system could see significant enhancements. One potential advancement involves integrating blockchain technology to streamline crop insurance claims, ensuring transparency and efficiency for farmers. Additionally, employing AI algorithms for advanced pest monitoring could enable early detection and targeted interventions to minimize crop damage. Developing AI-driven algorithms for smart crop rotation planning based on soil health data and historical performance could optimize yields and soil fertility. A user-friendly mobile application could provide personalized agronomic advice, weather forecasts, market trends, and training resources, empowering farmers with timely information. Furthermore, exploring hydroponic and vertical farming techniques integrated with IoT sensors and automation could boost production efficiency, reduce land requirements, and ensure year-round crop availability, particularly in urban areas. These advancements have the potential to revolutionize agricultural practices, making them more sustainable, efficient, and resilient.

CHAPTER – 8: BIBLIOGRAPHY

8.1 Books Referred

1. Roger S Pressman "Software Engineering-A Practitioner's Approach", Tata McGraw Hill 2004
2. "Internet of Things:A Hands-On Approach" by Arshdeep Bahga and Vijay Mandisetti,2015
3. "Introduction to Machine Learning with Python" by Andreas Muller
4. "Blockchain Basics: A Non-Technical Introduction in 25 Steps" by Daniel Drescher
5. "IoT Solutions in Microsoft's Azure IoT Suite: Data Acquisition and Analysis in the Real World" by Scott Klein, Mark Dunkel, and Paolo Patierno
6. "Practical Machine Learning for Computer Vision" by Valliappa Lakshmanan
7. "Blockchain Applications: A Hands-On Approach" by Arshdeep Bahga and Vijay Madisetti
8. "Precision Agriculture Technology for Crop Farming" edited by Qin Zhang
9. "Crop Modeling and Decision Support" edited by Jim Jones and Peter Antle

8.2 Websites Visited

1. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3949401/
2. https://www.instructables.com/Grow-With-CFL-Light-Experiment/
3. https://link.springer.com/article/10.1007/s00344-021-10337-y
4. https://www.frontiersin.org/research-topics/12923/crop-physiology-under-led-lighting
5. http://ecoursesonline.iasri.res.in/mod/page/view.php?id=1606
6. https://greenhouseemporium.com/blogs/greenhouse-gardening/greenhouse-gardening- how-to-grow-carrots/

8.3 References

[1] Akhila Susan Babu, Supriya M, "Blockchain Based Precision Agriculture Model Using Machine Learning Algorithms," 2022 International Conference on Breakthrough in Heuristics And Reciprocation of Advanced Technologies (BHARAT), October 2022.

[2] Hajar Moudoud, Soumaya Cherkaoui, Lyes Khoukhi, “An IoT Blockchain Architecture Using Oracles and Smart Contracts: the Use Case of a Food Supply Chain,” 2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), November 2019.

[3] M. Harini, D. Dhinakaran, D. Prabhu, S. M. Udhaya Sankar, V. Pooja, P. Kokila Sruthi, “Levarging Blockchain for Transparency in Agriculture Supply Chain Management Using IoT and Machine Learning,” 2023 World Conference on Communication & Computing (WCONF), September 2023.

[4] Ilhaam A. Omar, Raja Jayaraman, Khaled Salah, Haya R. Hasan, Jiju Antony, Mohammed Omar, “Blockchain-Based Approach for Crop Index Insurance in Agricultural Supply Chain,” IEEE Access (Volume: 11), October 2023.

[5] Kavita Saini, Ishika Mishra, Shreya Srivastava, “Farmer’s E-mart : An E-Commerce Store For Crops,” 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), March 2022.

[6] S. Madumidha, P. Siva Ranjani, U.Vandhana, B.Venmuhilan, “A Theoretical Implementation: Agriculture-Food Supply Chain Management using Blockchain Technology,” 2019 TEQIP III Sponsored International Conference on Microwave Integrated Circuits, Photonics and Wireless Networks (IMICPW), December 2019.

[7] Miguel Pincheira Caro, Muhammad Salek Ali, Massimo Vecchio, Raffaele Giaffreda, ”Blockchain-based Traceability in Agri-Food Supply Chain Management: A Practical Implementation,” 2018 IoT Vertical and Topical Summit on Agriculture - Tuscany (IOT Tuscany), June 2018.

[8] Ms. S. Madumidha, Dr. P. Siva Ranjani, Ms. S. Sree Varsinee, Ms. P.S. Sundari, ”Transparency and Traceability: In Food Supply Chain System using Blockchain Technology with Internet of Things,” 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), October 2019.

[9] H. Binici, M. Eken, M. Kara, M. Dolaz, ”An environment-friendly thermal insulation material from sunflower stalk, textile waste and stubble fibers,” 2013 International Conference on Renewable Energy Research and Applications (ICRERA), March 2014.

[10] Aditya Ghodke, Ajay Kokare, Rakesh Shinde, Akshay Marathe, P. S. Kashid, ”E-Application for Farmers to Sell Their Food Products through E-Auction”, https://api.semanticscholar.org/CorpusID:262077123, 2022.

[11] Aloysius Chiong Zhen Quan and Khairunnisa binti Sufian and Ng Sing Woei and Lai Li Ying and Amanda Ling Tzi Yun, ”FarmBid - A Bidding E-commerce Platform to Provide Fresh Produces from Farmers to Customers,” https://api.semanticscholar.org/CorpusID:244217697, September 2021

[12] Karthiga M, Srivarshan M, Danushmathi P, Sharmila S, ”BID.ITBIDDING AND E-AUCTION WEBSITE FOR FARMERS,” International Journal Of Advance Research And Innovative Ideas In Education Volume 9 Issue 4, 2023.

[13] Rahul Talreja, Rohan Chouksey, Sushma Verma, ”A Study of Blockchain Technology in Farmer’s Portal,” 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), September 2020.

[14] Mrs. Manisha Bhende, Ms. Mohini S. Avatade, Mrs. Suvarna Patil, Mrs. Pooja Mishra, Ms. Pooja Prasad, Mr. Shubham Shewalkar, ”Digital Market: E-Commerce Application For Farmers,” 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), April 2019.

[15] Apeksha Thorat, Sangeeta Kumari, Nandakishor D. Valakunde, ”An IoT Based Smart Solution for Leaf Disease Detection,” 2017 International Conference on Big Data, IoT and Data Science (BID), April 2018.

PATENT OFFICE
INTELLECTUAL PROPERTY BUILDING
G.S.T. Road, Guindy, Chennai-600032
Tel No. (091)(044) 22502081-84 Fax No. 044 22502066
E-mail : Chennai-patent@nic.in
Web Site : www.ipindia.gov.in

सत्यमेव जयते
GOVERNMENT OF INDIA

Docket Number:54487

Date/Time : 12/04/2024
Agent Number:

To,
KKR & KSR INSTITUTE OF TECHNOLOGY AND SCIENCES
VINJANAMPADU, VATTICHERUKU MANDAL, GUNTUR, ANDHRA PRADESH-522017. principaljr@gmail.com 9014156267

Sr No.	CBR No.	Reference Number /Application Type	Application Number	Title/Remarks	Amount Paid
1	25076	ORDINARY APPLICATION	202441029689	ENHANCED AGRICULTURAL MANAGEMENT APPROACH FROM SUSTAINABLE CROP PRODUCTION TO SELLING INCLUDING CROP	1750
2		E-2/2008/2024-CHE	202441029689	Form2	0
3	25076	E-12/4453/2024-CHE	202441029689	Form9	2750
4		E-106/4398/2024-CHE	202441029689	Form28	0
				Total :	**4500**

Received a sum of Rs. 4500 (Rupees Four Thousand Five Hundred only) through

Payment Mode	Bank Name	Cheque/Draft Number	Cheque/Draft Date	Amount in Rs
Draft	UCO Bank	366972	23/02/2024	4500

Note: This is electronically generated receipt hence no signature required.

Print CBR


madBlocks
Technology:Innovation:Business
INTERNSHIP CERTIFICATE
This is to certify that
Mr./Ms. DIVVELA SINDHU VENKATA DURGA GOWRI
student of the CSE department, KKR & KSR Institute of Technology and Sciences,
Guntur, Andhra Pradesh, has worked as an Engineering Intern on AI/IoT/Blockchain
Technology in the corporate engineering team from December 18th to April 18th 2024.
Intern ID: MS2024-122
Roll No: 20JR1A0523
We wish you all the best for your future.
MAKESKILLED
Innovations | Hackathons | Startups
CERTIFIED ISO 9001:2015 COMPANY
Microsoft Partner
MADHU PARVATHANENI
Founder & CEO,
Madblocks Technologies Pvt Ltd
Make Skilled


madBlocks
Technology:Innovation:Business
INTERNSHIP CERTIFICATE
This is to certify that
Mr./Ms. CHATHARASUPALLI GAYATHRI
student of the CSE department, KKR & KSR Institute of Technology and Sciences, Guntur, Andhra Pradesh, has worked as an Engineering Intern on AI/IoT/Blockchain Technology in the corporate engineering team from December 18th to April 18th 2024.
Intern ID: MS2024-110
Roll No: 20JR1A0510
We wish you all the best for your future.
MAKESKILLED
Innovations | Hackathons | Startups
Microsoft Partner
MADHU PARVATHANENI
Founder & CEO,
Madblocks Technologies Pvt Ltd
Make Skilled


madBlocks
Technology:Innovation:Business
INTERNSHIP CERTIFICATE
This is to certify that
Mr./Ms. JAGARLAMUDI SAI SATHVIKA
student of the CSE department, KKR & KSR Institute of Technology and Sciences, Guntur, Andhra Pradesh, has worked as an Engineering Intern on AI/IoT/Blockchain Technology in the corporate engineering team from December 18th to April 18th 2024.
Intern ID: MS2024-127
Roll No: 20JR1A0528
We wish you all the best for your future.
MAKESKILLED
Innovations | Hackathons | Startups
Microsoft Partner
MADHU PARVATHANENI
Founder & CEO,
Madblocks Technologies Pvt Ltd
Make Skilled


AICTE
All India Council for Technical Education
madBlocks
Technology:Innovation:Business
INTERNSHIP CERTIFICATE
This is to certify that
Mr./Ms. MATHE VIJAYA MERCY
student of the CSE department, KKR & KSR Institute of Technology and Sciences, Guntur, Andhra Pradesh, has worked as an Engineering Intern on AI/IoT/Blockchain Technology in the corporate engineering team from December 18th to April 18th 2024.
Intern ID: MS2024-226
Roll No: 21JR5A0504
We wish you all the best for your future.
MAKESKILLED
Innovations | Hackathons | Startups
CERTIFIED ISO 9001:2015 COMPANY
Microsoft Partner
MADHU PARVATHANENI
Founder & CEO,
Madblocks Technologies Pvt Ltd
Make Skilled