Title of the Project

Agro-Health

Course Code: CIS499

Fall-2023

Supervised By:

Syed Tangim Pasha

Lecturer, Department of CIS Daffodil International University

Submission By:

Monjur E Alahi Chowdhury

ID: 201-16-501

CIS 8th batch

Submission Date: 23 January 2024

APPROVAL

This Project titled "AgroHealth", Submitted by Monjur E Alahi Chowdhury, ID No:201-16-501 to the Department of Computing & Information Systems, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computing & Information Systems and approved as to its style and contents. The presentation has been held on 23-01-2024.

BOARD OF EXAMINERS

23/01/24

Mr. Md Sarwar Hossain Mollah
Associate Professor and Head
Department of Computing & Information Systems
Faculty of Science & Information Technology
Daffodil International University

Chairman

Dr. Shekh Abdullah-Al-Musa Ahmed
Assistant Professor
Department of Computing & Information Systems
Faculty of Science & Information Technology
Daffodil International University

Internal Examiner

Md. Nasimul Kader
Assistant Professor
Department of Computing & Information Systems
Faculty of Science & Information Technology
Daffodil International University

Internal Examiner

Prof. Mohammad Shorif Uddin
Professor
Department of Computer Science and Engineering
Jahangirnagar University

External Examiner

DECLARATION

I hereby declare that; this thesis has been done by me under supervision of Syed Tangim Pasha, Lecturer, department of Computing and Information System (CIS) of Daffodil International University. I am also declaring that this thesis or any part of there has never been submitted anywhere else for the award of any educational degree like, B.Sc., M.Sc., Diploma or other qualifications.

Supervised By

23.1.24

Syed Tangim Pasha
Lecturer
Department of CIS
Daffodil International University

Submitted By

Monjur

Name: Monjur E Alahi Chowdhury
ID: 201-16-501
Department of CIS
Daffodil International University

ACKNOWLEDGEMENT

I give thanks to Allah for enabling the effective completion of this task. I would want to thank Syed Tangim Pasha, a lecturer in computing and information systems, for being my amazing supervisor. His constant guidance and assistance throughout the project were crucial in giving the confidence required to take it through to completion. The Department of Computing and Information Systems at Daffodil International University deserves recognition for promoting a learning environment and contributing to the project's success. Throughout my bachelor's degree, I would like to thank all of my diligent lecturers, especially the Head of the Department of Computing and material Systems, for contributing material that was crucial to the project's progress. The information gained from many classes served as the foundation for this endeavor. I used a range of sources, including books, journals, electronic media, and other secondary references, to gather important information for my research. Their contributions made the project richer and more in-depth. I am grateful for my friends' unwavering support in fostering a cheerful and happy environment. Their companionship helped them overcome many challenges. Lastly, I would want to sincerely thank everyone who contributed to this project. Every effort, no matter how tiny, has shaped its success, and for that, I am very appreciative.

DEDICATION

In addition to my parents and family, who have all helped me to finish my BSC program, this project is dedicated to my Creator. I also give thanks to Almighty God for his guidance, protection, and support. Additionally, I would want to dedicate my effort to my seniors and pals who are truly assisting me in maturing, as well as to my esteemed professors who have honestly assisted me in being competent.

ABSTRACT

Presenting "AgroHealth," a cutting-edge online application designed to satisfy the many demands of farmers and medical professionals working in the agriculture industry. This dynamic platform aims to improve the general health and well-being of crops by acting as a central point for smooth communication and collaboration. With the help of AgroHealth's user-friendly interface, farmers may effectively report and record health-related problems impacting crops. A consolidated database with vital details about agricultural circumstances, illness histories, and particular health needs is established via the online program. Healthcare workers may make well-informed judgments and offer prompt support with the help of this repository. One of AgroHealth's best features is its live video calling feature, which lets users find the closest agricultural health centers. This makes sure that health-related issues are routed quickly and accurately, linking farmers to the closest medical providers. Moreover, AgroHealth delivers real-time updates and notifications, keeping farmers and healthcare professionals informed about crucial concerns and trends in their agricultural community. This function improves the industry's general awareness and reactivity. With the use of AgroHealth's collaboration tools, farmers and medical experts can communicate more effectively and share vital information including immunization schedules, disease management plans. This simplified method improves agricultural health practices and raises the general health of crops and cattle.

Table of Contents

CHAPTER 1: INTRODUCTION

1.1 Introduction

Introducing "AgroHealth" - a revolutionary online application designed to meet the many and pressing demands of farmers and medical professionals working in the agriculture industry. AgroHealth is a shining example of how agriculture can use digital innovation to improve the health and vitality of crops and livestock in a time when the industry is at the forefront of sustainability and technology. A farmer's job is to protect their harvests, yet the agricultural environment is full with obstacles, from insect incursions to unforeseen illnesses. Providing a dynamic platform for effective communication and cooperation, AgroHealth enters this market as a complete solution. It is more than simply a web application; it is an essential link between farmers and medical professionals that helps to build a stronger, healthier agricultural ecosystem. The core of AgroHealth is its intuitive interface, which enables farmers to easily report and record health-related problems impacting their crops. The program creates a centralized database that serves as a goldmine of data, including vital information on farming conditions, illness histories, and particular health needs. For healthcare workers, this library serves as their skeleton, enabling them to make deft judgments and provide prompt support. AgroHealth's live location feature is one of its best qualities. Farmers may quickly locate veterinary clinics, emergency rooms, and agricultural health centers in the area, guaranteeing prompt and accurate routing of health-related issues. This helps farmers get in touch with the closest medical providers while also saving them a significant amount of time. Real-time updates and notifications inside AgroHealth keep farmers and healthcare professionals in the loop about crucial concerns and changes within their agricultural community. The agriculture industry is made more responsive and integrated by this increased awareness. AgroHealth is really a collaborative force that is bringing agriculture into the modern century rather than only being a web application. Through the integration of technology, effective communication, and prompt health support, AgroHealth seeks to strengthen the foundation of agriculture and guarantee a more sustainable and healthy future for both crops and cattle.

CHAPTER 2: INITIAL STUDY

2.1Project Proposal

Objective:

The main objective of the "AgroHealth" project is to develop and implement an integrated web application that addresses the challenges faced by farmers in Bangladesh. The system aims to provide comprehensive solutions through innovative technologies and services, facilitating a more efficient and sustainable agricultural environment.

Features:

- ✓ Provide a facility search function in AgroHealth that is focused on farmers for convenient access. Include filters to deliver results that are tailored to each person's requirements and tastes.
- ✓ Provide an adaptable service access mechanism in AgroHealth so that farmers may communicate via phone calls and AI chatbots. Assure a flawless user experience that accommodates a range of access levels and preferences for technology.
- ✓ For increased security, use Stripe and SSL to strengthen AgroHealth's payment transactions. Enhance the payment gateway to enable bitcoin transactions, guaranteeing farmers a transparent and safe financial environment.
- ✓ Create an anticipatory AgroHealth system that contacts and texts farmers with offers and updates, strengthening ties within the community.
- ✓ Simplify AgroHealth's service booking procedure so that farmers may more easily obtain necessary services thanks to an enhanced, user-friendly interface.

- ✓ Facilitate contacts between farmers and experts in the field of agrohealth. Encourage advice-seeking by utilizing tools such as live chats and discussion forums to exchange information.
- ✓ Give AgroHealth's AI Chatbot the tools it needs to quickly and accurately handle farmers' problems in the moment by using machine learning to make constant improvements in accuracy and responsiveness.
- ✓ To remain ahead of possible risks, make sure that AgroHealth has strong data security, protecting the privacy of farmers through frequent upgrades and evaluations.
- ✓ Diversify payment options in AgroHealth beyond blockchain, incorporating popular methods like credit cards and digital wallets, providing farmers flexibility in choosing their preferred payment mode.

Benefits:

- ✓ Immediate Solutions for Agricultural Issues
- ✓ Involvement with the Community and Knowledge Exchange
- ✓ Simple Service Booking and Access
- ✓ Flexible and Secure Financial Transactions
- ✓ Assurance of Privacy and Confidentiality for Farmers

Timeline and Resources:

The installation of AgroHealth will be conducted through a methodical tiered approach, comprising design, development, testing, and deployment stages. For the project to be implemented successfully, a skilled team of developers, designers, and project managers will be essential.

Conclusion:

AgroHealth is a cutting-edge approach that has the potential to completely transform the agriculture industry. The project aims to improve the productivity, communication, and general well-being of the agricultural community by utilizing a user-centric approach, strong security measures, and features that empower farmers.

AgroHealth hopes to bring in a new age of prosperous and sustainable agriculture by embracing cutting-edge technologies and encouraging teamwork.

2.2 Background of the AgroHealth

A strong framework created to promote ongoing contact between farmers, agricultural specialists, and service providers is the foundation of the AgroHealth program. A web application is necessary to do this while also meeting the various needs of consumers. Three main components make up the system: Admin, Users, and Agricultural Experts. A comprehensive agricultural solution is provided by the convergence of many functional characteristics within this comprehensive framework. These include the ability to add users and agriculturist, as well as to facilitate real-time video calling, volunteer possibilities. The Admin module is the most powerful module in the system. It is responsible for handling important functions including adding new agriculturist and users and controlling factors relating to agriculture. Within their module, users have access to a variety of features, such as managing their profiles, interacting directly with agriculturist, and postings about their problem. A sustainable ecology is ensured via financial transactions, which are an essential component of the user experience. AgroHealth appears as a game-changing initiative that unifies services, information sharing, and communication to foster a robust and prosperous farming community. The proposal envisions an agricultural environment that is connected and empowered via the seamless integration of technology.

2.3 Problem Area

Data Integrity and Accuracy: It can be difficult to keep agricultural data accurate, including crop health reports, and service availability. Maintaining data integrity requires routine system maintenance and verification procedures.

User Adoption and Engagement: It could be difficult to get farmers to use the AgroHealth platform's capabilities and actively interact with it. User adoption may

be improved by running awareness campaigns, emphasizing the advantages of the platform, and encouraging user education.

Integration and Compatibility: There may be technological challenges in integrating AgroHealth with current agricultural systems and guaranteeing compatibility across a range of devices and operating systems. For a flawless user experience, these compatibility concerns must be resolved.

Privacy and Security: Strong security measures are needed when handling sensitive agricultural data, such as crop statistics, pest reports, and farmer details. Protecting agricultural data requires the use of encryption, secure authentication, and adherence to privacy laws.

Collaboration and Resource Availability: Ensuring sufficient resources, including expert availability, agricultural service capacity, and tool access, is essential to fulfilling demands on the AgroHealth platform. Resource-related issues can be addressed by forming alliances and partnerships with pertinent agricultural groups.

User Interface and User Experience: Creating an easy-to-use interface that is nice to farmers is crucial to their satisfaction. Any usability issues with the AgroHealth platform may be found and fixed during the development phase by doing user testing and collecting feedback.

Regulatory Compliance: It's critical to follow local agriculture rules and regulations. It will be essential to stay up to date on legal requirements and make sure AgroHealth complies with pertinent rules in order to prevent legal complications and guarantee the project's success.

2.4 Possible Solution

The following are possible remedies to problems found throughout the AgroHealth project's development:

Data Integrity and Accuracy: Within the AgroHealth platform, establish a methodical procedure for routine data updates and verification. Form alliances with trustworthy agricultural sources to guarantee accurate and up-to-date information on

crop health, weather, and service accessibility. Include user feedback systems so that farmers may report any outdated or inaccurate information.

User Adoption and Engagement: Employ focused marketing techniques to increase the agricultural community's knowledge of AgroHealth and its advantages. Offer discounts on farming equipment or exclusive access to instructional content as incentives or prizes for active platform involvement. Work together with regional agricultural associations to raise awareness of AgroHealth and motivate members to take an active role in it.

Integration and Compatibility: To guarantee a smooth integration with current agricultural systems and meet the various demands of farmers, carry out thorough testing and quality assurance. For broader use, develop AgroHealth utilizing cross-platform frameworks or make sure it is compatible with all main operating systems. Provide consumers with thorough instructions and assistance in troubleshooting device compatibility difficulties.

Privacy and Security: To protect sensitive agricultural data inside AgroHealth, use cutting-edge data encryption techniques. Enforce stringent adherence to privacy standards by using robust authentication procedures and accurately managing permissions. Update security protocols often, and carry out audits to find and fix possible weaknesses.

Availability of Resources and Collaboration: To build a strong network of resources inside the AgroHealth platform, cultivate partnerships with veterinary clinics, service providers, and agricultural organizations. Incorporate user feedback methods to provide real-time resource availability reports, improving information accuracy. To boost participation in agricultural programs and support services, promote farmer involvement and offer incentives.

User Interface and Experience: To find and fix any usability issues, thoroughly test the product with users and collect their input throughout the AgroHealth development process. Use design ideas that are intuitive to ensure a smooth and user-friendly experience for farmers. Provide farmers with in-app training or guidelines to help them navigate and use AgroHealth efficiently.

Regulatory Compliance: Make sure AgroHealth complies with all applicable local agricultural rules and regulations by keeping up with them. Work together with legal counsel to ensure continued compliance, and quickly update the

platform to take into account any changes to regulations. By putting these fixes into practice, AgroHealth will be able to go over any obstacles and provide farmers with a useful tool for their farming requirements.

CHAPTER 3- LITERATURE REVIWE

3.1 Discussion on problem domain based on published articles

Present-day agricultural applications frequently concentrate on a single topic, forcing users to search across several platforms in search of all-encompassing answers. The disorganized nature of these apps results in a disjointed user experience. By combining several agricultural aspects into a single, well-organized platform, AgroHealth fills this gap and offers farmers a comprehensive, user-friendly solution for their range of requirements.

3.2 Talk about how to solve problems using published papers as a basis.

- ✓ Farmer-Centric Design: Create AgroHealth from the perspective of the farmer, putting the user's experience first with a user-friendly interface and smooth navigation.
- ✓ Classify Features: For ease of access, group AgroHealth features into discrete areas like Crop Health, Weather Updates, Agricultural Services, and Pest Management.
- ✓ Welfare of Farmers The relevance of farmer well-being should be emphasized via the integration of educational resources, mental health care, and partnerships with agricultural groups.
- ✓ Accessibility of Agricultural Resources: Work with suppliers, provide a variety of agricultural resources, provide subscription services, and coordinate donations for resource support.

By using these strategies, AgroHealth hopes to improve farmers' experiences in agriculture overall by streamlining its features and emphasizing the value of resource accessibility and farmer well-being

3.3 Comparison of the top three or four solutions

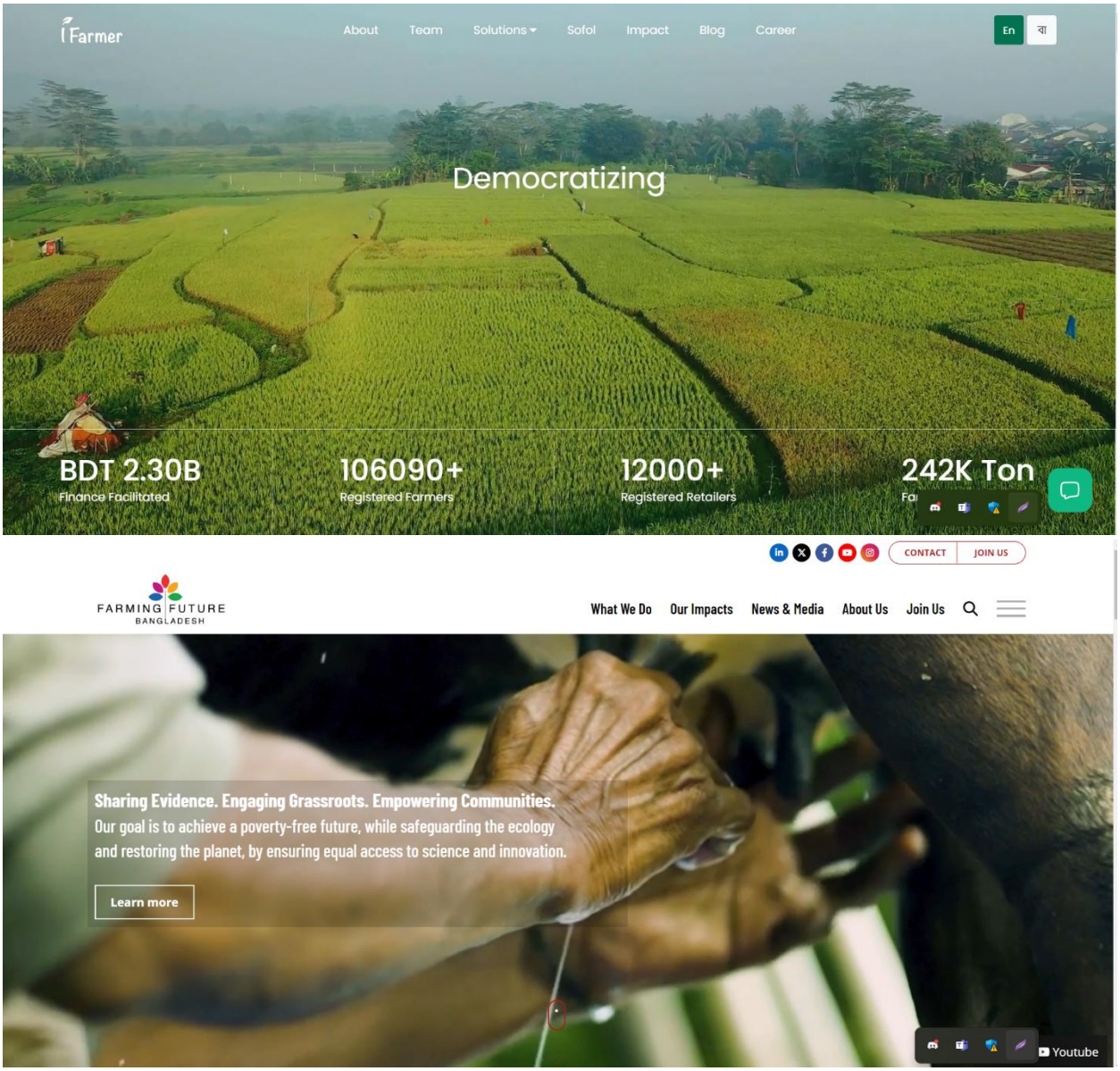


Figure 1: some app collecting from Google

Best Features:

- ✓ Reservation and Refund of Services

- ✓ Interactive Forums for Agriculture
- ✓ AI-powered Issue Resolution
- ✓ All-inclusive Agricultural Services
- ✓ In-person video consultation
- ✓ Guaranteed security
- ✓ Cross-Device Interoperability
- ✓ Sensational Navigation
- ✓ Easy-to-use interface
- ✓ User Interfaces according to roles

Limitations:

- ✓ Agricultural data shown graphically
- ✓ No banking or card payment system
- ✓ Messaging App
- ✓ Module for Team Collaboration

3.3 Recommended approach

Farmers' Panel

- ✓ Login/Logout
- ✓ Dashboard
- ✓ Crop Monitoring System
- ✓ Agricultural Services Booking
- ✓ Consultation with Agricultural Experts
- ✓ AI-driven Problem Solving
- ✓ Direct Video Consultation

Admin panel

- ✓ Login/Logout
- ✓ User Verification
- ✓ Add Agricultural Experts
- ✓ Service Approval
- ✓ Security Measures
- ✓ Dashboard Analytics
- ✓ Regulatory Compliance

Agricultural Experts' Panel

- ✓ Login/Logout
- ✓ Expert Consultation Module
- ✓ Request Notifications
- ✓ Interactive Forums
- ✓ Interact with farmers via message and video call

CHAPTER 4: METHODOLOGY

4.1 What to use

This chapter describes approach used to achieve the project goals in the AgroHealth program. The Software Development Lifecycle (SDLC), a thorough framework for software design, development, and maintenance, is at the heart of the selected methodology. AgroHealth will take into account well-known models like Agile, Waterfall, V Model, Spiral Model, and Dynamic System Development Model (DSDM) among others that are part of the software development process. Each model provides an organized and adaptable framework to direct the creation, execution, and upkeep of the AgroHealth platform. It does this by offering unique procedures, stages, benchmarks, and techniques for monitoring progress. In order to provide an efficient and methodical development process that is in line with the aims and objectives of the AgroHealth project, the chosen SDLC model will be customized to the particular requirements of the agricultural domain.

4.2 Why to use

AgroHealth uses the Dynamic System creation Model (DSDM) with a greater emphasis on projects than most other agile techniques, stressing the creation and delivery of agricultural products rather than just software. Considering the numerous components required to fulfill the broader context of agricultural solutions, DSDM meets bigger business demands, given the diverse and complicated nature of agricultural requirements. DSDM has a long history of delivering agile projects effectively, and it has proven to be scalable and effective in both small and straightforward organizations. Its adaptability goes beyond IT projects; it has demonstrated success in non-IT fields as well, such corporate transformation programs in the agriculture industry. AgroHealth is an example of this methodology's successful implementation in the field of agriculture, and it acts as a testimonial to its usefulness.

4.3 Methodology section

Pre Project Phase:

- ✓ **Feasibility Study:** Assess the AgroHealth project's viability while taking operational, financial, and technological factors into account. Examine the advantages, disadvantages, and possible hazards of the agricultural platform.
- ✓ **Conditions Collecting:** Compile and record AgroHealth's unique requirements, emphasizing user expectations and agricultural demands. Recognize limitations and specify the entire agricultural platform's reach.
- ✓ **Planning:** Create a thorough project plan for AgroHealth that includes deliverables, deadlines, resource needs, and goals. Establish communication and risk management strategies, specify roles and duties, and list project stakeholders for the agriculture platform

Project Lifecycle Phase:

- ✓ **Design:** Using the information acquired about agricultural needs, create the AgroHealth design. Manage high-level and in-depth design tasks, including database design, user interface design, and architectural design.
- ✓ **Development:** Utilize the design guidelines to code the AgroHealth platform. Carry out component integration and unit testing to create a working software product for farming.
- ✓ **Testing:** To guarantee AgroHealth's quality and functionality in the agricultural arena, do thorough testing on the product. Carry out several kinds of testing, such as system, user acceptability, integration, and unit testing.
- ✓ **Deployment:** Following extensive testing and approval, deploy AgroHealth to the production environment. Oversee the agricultural platform's installation, setup, and configuration in the intended setting.

Post-Project Phase:

- ✓ **Maintenance:** AgroHealth enters the maintenance phase following deployment, during which time it focuses on upgrades, bug repairs, and ongoing support. Make sure it continues to work and can adjust to changing needs in agriculture.
- ✓ **Evaluation:** Analyze the AgroHealth project's effectiveness by contrasting its actual results with its intended goals. Determine what needs to be improved upon and compile lessons gained to guide future farming endeavors.
- ✓ **The release:** Complete the formal closing step of the AgroHealth project. Complete all project documentation, carry out a thorough post-mortem or review, and save project artifacts for future use and understanding.

These parts cover everything from pre-implementation planning to post-implementation assistance, offering an organized method for managing software development projects and assisting in achieving successful results.

4.4 Implementation plans

At this pivotal point in the AgroHealth project's development, the finished agricultural platform is made available to all users. The system's usability is given top priority, therefore any errors found are fixed quickly to provide a flawless experience. The strategy for implementation includes the choice of environments, methods, and criteria for release that are customized to meet the particular needs of the agriculture industry. The emphasis during the testing and rollout of the new system is still on quick fixes for any problems found, creating an atmosphere that allows the AgroHealth platform to be improved and optimized over time for the good of its agricultural community users.

CHAPTER 5: PLANNING

5.1 Project Schedule

A thorough project Schedule that outlines AgroHealth project scope, cost, timetable, risk, resources, and communication techniques serves as a roadmap for its execution and monitoring. Thorough planning is done prior to implementation in order to minimize risks and guarantee a seamless transition. These materials describe every aspect of the AgroHealth initiative and act as guides. Tasks like goal formulation, risk reduction, and deadline adherence are addressed in this planning, which benefits the project manager, team, and participants. The creation of a Work Breakdown Structure (WBS), time boxing, and Gantt charts is part of the project plan; these tools are essential for organizing activities and deadlines in the AgroHealth initiative in a systematic manner. Some examples of these are:

5.1.1 Management Plan

Describe the project team's duties and responsibilities as well as the project management methodology. To guarantee successful cooperation, set up the reporting and communication routes. Determine the issue-resolution process's decision-making steps and escalation protocols.

No	Task	Duration	Start Date	End Date
1	Introduction	5	1 Jun 2023	5 Jun 2023
2	Initial Study	4	6 Jun 2023	9 Jun 2023
3	Literature Review	4	10 Jun 2023	13 Jun 2023
4	Methodology	3	14 Jun 2023	16 Jun 2023
5	Planning	10	17 Jun 2023	26 Jun 2023
6	Feasibility	15	27 Jun 2023	13 Jul 2023
7	Foundation	5	14 Jul 2023	18 Jul 2023
8	Exploration	14	19 Jul 2023	3 Aug 2023
9	Engineering	30	4 Aug 2023	27 Aug 2023
10	Deployment	18	28 Aug 2023	15 Sep 2023
11	Testing	10	16 Sep 2023	25 Sep 2023
12	Implementation	5	26 Sep 2023	30 Sep 2023
13	Critical Appraisal and Evaluation	4	2 Oct 2023	5 Oct 2023
14	Lessons Learning	3	6 Oct 2023	8 Oct 2023
15	Conclusion	1	9 Oct 2023	10 Oct 2023
	Total	94 days		

Table 1: Management Planning

5.1.2 Allocation of Resources

Determine all the resources needed for the project, such as staff, tools, and software. Adapt resource allocation to the workload and schedule of the project. Assign team members duties and responsibilities and make sure they possess the required knowledge and abilities.

No	Task Name	Duration	Resource
1	Introduction	5	End User
2	Initial Study	4	Analyst
3	Literature Review	4	Analyst
4	Methodology	3	Analyst
5	Planning	10	Analyst, Designer, Developer
6	Feasibility	15	Analyst
7	Foundation	5	Designer
8	Exploration	14	Designer, Developer
9	Engineering	30	Developer
10	Deployment	18	Analyst, Developer
11	Testing	10	Analyst, Developer, Tester, Users
12	Implementation	5	Analyst, Developer
13	Critical Appraisal and Evaluation	4	Analyst, Tester and Developer
14	Lessons Learning	3	Analyst, Users
15	Conclusion	1	Analyst
	Total	94 days	

Table 2: Resource allocation

5.1.3 Time Boxing

To make development and testing easier, divide the project into discrete time frames or iterations. Determine the length of each time box as well as the assignments and outputs that must be finished in each iteration. For every time box, assign resources and establish distinct objectives.

Time Box	Task Name	Duration	Resource
TB1	Introduction	5	End Users, Analyst
	Study	4	Analyst
	literature review	4	Analyst
TB2	Methodology	3	Analyst
	Planning	10	Analyst, Designer, Developer
	Feasibility	15	Analyst
TB3	Foundation	5	Designer
TB4	Exploration	14	Designer, Developer
	Engineering	30	Development
TB5	Deployment	18	Analyst, Developer
	Testing	10	Analyst, Tester, Users
TB6	Implementation	5	Developer
TB7	Critical Appraisal and Evaluation	4	Tester and Developer
	Learning	3	Users
TB8	conclusion	1	Analyst
	Total	94 days	

Table 3: Time Boxing

5.1.4 Gantt Chart

Gantt charts play a crucial role in project management as they help navigate the intricacies of challenging projects, increasing the probability of successful completion for AgroHealth. Using Gantt charts makes planning and scheduling more efficient and makes coordinating jobs of different sizes easier. These charts clarify several plan kinds, set deadlines for each system phase, and visually depict the amount of work to be done at each step using horizontal bars. With the ability to show the beginning and ending dates of each job, the Gantt chart proves to be a useful tool in AgroHealth, providing a clear and complete picture that facilitates effective management and the successful completion of the agricultural endeavor.

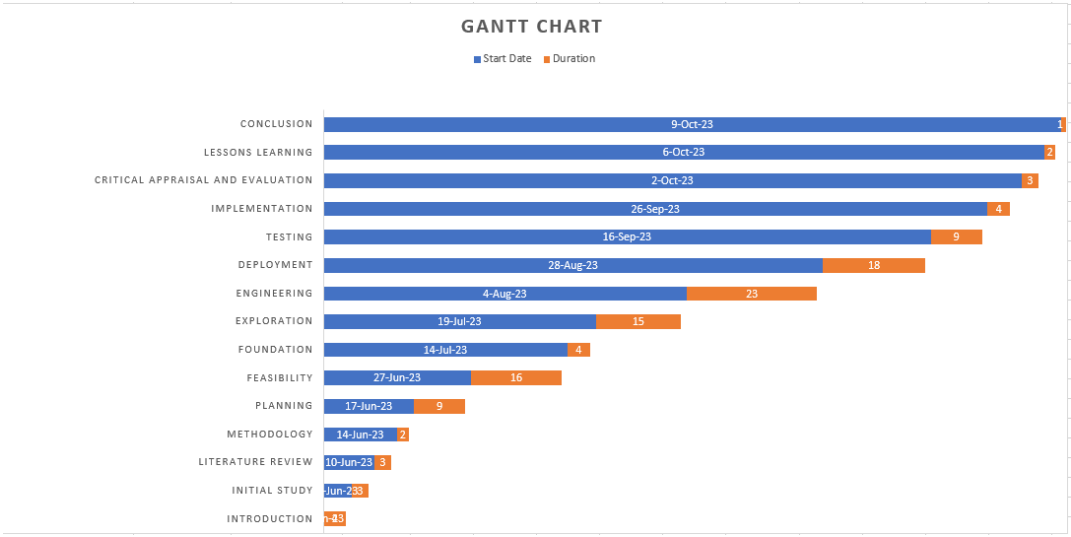


Figure 2: Gantt chart

CHAPTER 6: FEASIBILITY

AgroHealth project execution requires a comprehensive approach that takes into account engineering, planning, legal, and economic aspects. A crucial success indicator is feasibility analysis. A project's capacity to supply agricultural goods to clients, cost, and return on investment (ROI) are important considerations when determining the possibility of success. AgroHealth's feasibility analysis encompasses a thorough assessment that guarantees the coherence of engineering tactics, compliance with regulations, and financial sustainability. The project's ability to successfully supply end users with agricultural solutions is a critical determinant of its success, highlighting the significance of consumer acceptability and product viability in the agricultural sector.

6.1 Every Potential Kind of Feasibility

6.1.1 Operational viability

Making use of AgroHealth is a simple process that can be done from any place with an internet connection. The web-based program simplifies work for agriculturist, farmers, and consumers by guaranteeing cost and ease of use. The idea makes it simple to adapt with little equipment—just a computer and an internet connection—by getting rid of repetitive chores and encouraging transparency. Having all departments integrated allows for smooth communication and improves overall productivity. AgroHealth is a workable option that offers agricultural stakeholders a useful, networked, and user-focused platform that greatly enhances their operational procedures.

Technical Feasibility

Hardware	Software	Database
HP Laptop, Wi-Fi, Router, Cable.	VS Code, Google Chrome Browser, Windows, MS Word, Firebase	MongoDB

6.1.2 Technology

Client Side	Server Side
React JS	Node JS

6.2 Cost Benefit Analysis

The cost-benefit analysis is a crucial tool in project management as it helps evaluate the advantages and disadvantages of different project paths that include transactions, activities, investments, and business demands. There are a number of possibilities within the AgroHealth project, and the cost-benefit analysis provides a framework for determining the best course of action for accomplishing project objectives while minimizing costs. This analytical method guarantees that the AgroHealth program carefully distributes resources, optimizing benefits while reducing expenses. AgroHealth's cost-benefit analysis turns into a crucial instrument for decision-making, pointing the project in the direction of practical and affordable approaches to benefit agricultural stakeholders.

Project Name: AgroHealth

Serial	Tools	First	Second	Third	Fourth	Total
1	Cost of Web base Project	৳ 1,04,000	-	-	-	৳ 1,05,000
2	Cost of desktop base Project	৳ 1,02,000	-	-	-	৳ 1,02,000

3	The cost of domain plus hosting	৳25,000	৳25,000	৳25,000	৳35,000	৳ 100,000
4	Expenses for Employees	৳40,000	৳40,000	৳40,000	৳40,000	৳1,60000
5	Other Cost	৳20,000	৳20,000	৳20,000	৳20,000	৳8,0000
6	Total Cost	৳ 291,000	৳ 85,000	৳ 85,000	৳ 95,000	৳ 547,000

Table 4: Cost Benefit

DSDM Dynamic System Development Method (DSDM)

AgroHealth's changing needs are easily aligned with the vendor-independent and scalable Dynamic System Development Method (DSDM), which guarantees on-time and within-budget project delivery. DSDM is an appropriate technique for AgroHealth's agricultural technology development and deployment because of its agile base and vendor-independence.

CHAPTER 7: FOUNDATION

7.1 Some potential approaches

7.1.1 Interviews

Interview producers, specialists in agriculture, and recipients of agricultural services. Examine their problems, regions of discomfort, and places that need to be improved in the agricultural field. Find more about challenges in managing agricultural

operations, organizing farming activities, and obtaining manpower. Learn about their requirements, communication barriers, and problems with resource distribution.

7.1.2 Observations

To watch farming operations, visit farms, agricultural centers, or other relevant institutions. Examine how planting, harvesting, and other agricultural operations are carried out. Determine any inefficiencies, bottlenecks, or places where manual procedures might be made more efficient. Note difficulties with data monitoring, information exchange, and general farming process management.

7.1.3 Questionnaires

Give out questionnaires to potential AgroHealth app users, farmers, and agricultural specialists. Get input on their opinions of the current systems, their degree of satisfaction, and potential areas for development. Ask them about qualities they value and what they would anticipate from an app that provides agricultural aid. Ask for feedback on how to make it easier to obtain information, get advice on agriculture, and enhance farming methods. These methods will offer insightful information for setting feature priorities and creating solutions for problems facing the agriculture industry.

7.2 Specific problem area identification and description

Inadequate coordination and communication among farmers, leading to misunderstandings, delays, and inefficiencies in farming operations as a whole. This disrupts the cultivation cycle and the smooth transfer of farming resources between farmers, impeding timely and suitable agricultural operations.

7.3 Possible Solution

Develop an integrated AgroHealth app that will act as a focal point for the agricultural community's effective coordination, communication, and information

exchange. Real-time communication channels, centralized profiles for farming resources, agricultural practice management tools, and labor and resource coordination features should all be included in the app. The objective of this solution is to improve stakeholder communication, optimize farming procedures, and raise the general efficacy and efficiency of agricultural operations.

7.4 List of All Requirements There are two categories of criteria.

1. Functional
2. Non-Functional

Functional Requirements

- ✓ Admin has to sign in and out.
- ✓ Admin needs to confirm user identities.
- ✓ Agricultural specialists can be added by the admin.
- ✓ Resource management is possible for admin.
- ✓ Admin is able to control orders and inventory in the marketplace.
- ✓ Farmers need to sign in and out.
- ✓ Farmers need to have access to the Dashboard.
- ✓ Farmers have to disclose their farming locations.
- ✓ Farmer use of the paid and booked system is required.
- ✓ In case of an emergency, farmers must contact.
- ✓ Agricultural services are available to farmers.
- ✓ Resources for farming can be bought by farmers.
- ✓ Experts in agriculture need to log in and out.
- ✓ Experts in agriculture need their own module.
- ✓ Requests for consultation must reach agricultural specialists.

Non-Functional Requirements

- ✓ User-friendly system for simple navigation.
- ✓ robust security protocols to safeguard user and agricultural data.
- ✓ Guarantee the privacy and accuracy of personal agricultural data.
- ✓ Make reports and summary easily readable.
- ✓ On the platform, only show the information that farmers need to see.

- ✓ Ensure responsiveness on all sorts of devices and browsers.

7.5 Which Technology to be implemented

For AgroHealth, selecting the MERN (MongoDB, Express.js, React, Node.js) stack might be a wise choice for the following reasons:

- ✓ **Cross-platform development:** By enabling the creation of online and mobile apps, the MERN stack guarantees a unified user experience on a range of platforms.
- ✓ **Quick and effective development:** The MERN stack's Node.js component allows server-side JavaScript execution, promoting a single language and codebase and expediting the development process.
- ✓ **Performance:** The asynchronous, event-driven design of Node.js is well-known for offering excellent performance and scalability. Faster response times are the outcome, which is necessary for a user interface that is responsive.
- ✓ **Rich user interface and customizability:** React, a fundamental part of the MERN stack, provides an architecture based on components, which facilitates the creation of dynamic and interactive user interfaces. Reusable parts are supported, which improves maintainability and customization.
- ✓ **Expanding ecosystem and community**: React boasts a large and vibrant community that guarantees an abundance of tools, libraries, and community support. Development is accelerated by this ecosystem because common problems have easy access to solutions.
- ✓ **JavaScript (React) language:** One of the most popular programming languages is JavaScript (React), which streamlines user interface creation by using a declarative methodology. This makes the code easier to understand and update.
- ✓ **Strong backend support:** The MERN stack's Express.js component acts as a strong backend framework. It handles HTTP requests effectively and makes API building simpler. The NoSQL database MongoDB guarantees scalability and flexibility, allowing for the changing nature of agricultural data.
- ✓ **A single JavaScript-based development environment is provided by the full-stack JavaScript:** MERN stack, which enables developers to easily transition between frontend and backend operations. This consistency lowers the learning curve and improves teamwork.

Though the final decision about technology selection is based on team composition, project needs, and scalability issues, the MERN stack is notable for its adaptability, developer-friendliness, and capacity to produce high-performing and scalable applications for the AgroHealth project.

7.1 Recommendations and Justifications

AgroHealth's web application was developed using an efficient technological stack that was carefully selected to fulfill certain needs. The backend framework, Express.js, gave server-side programming a stable and expandable base. A NoSQL database called MongoDB was used to effectively store and handle data, providing agricultural data handling flexibility and scalability.

React was used on the front end to provide a responsive and dynamic user experience. The component-based design of React made it easier to create modular and reusable user interface components, which enhanced the interaction and interest of the user experience. This technological stack, which combines Express.js, MongoDB, and React, guarantees a comprehensive and effective solution for AgroHealth by tackling database administration, frontend user interaction, and backend functionality in a unified and integrated manner.

CHAPTER 8: EXPLORATION

8.1 Old System Use Case

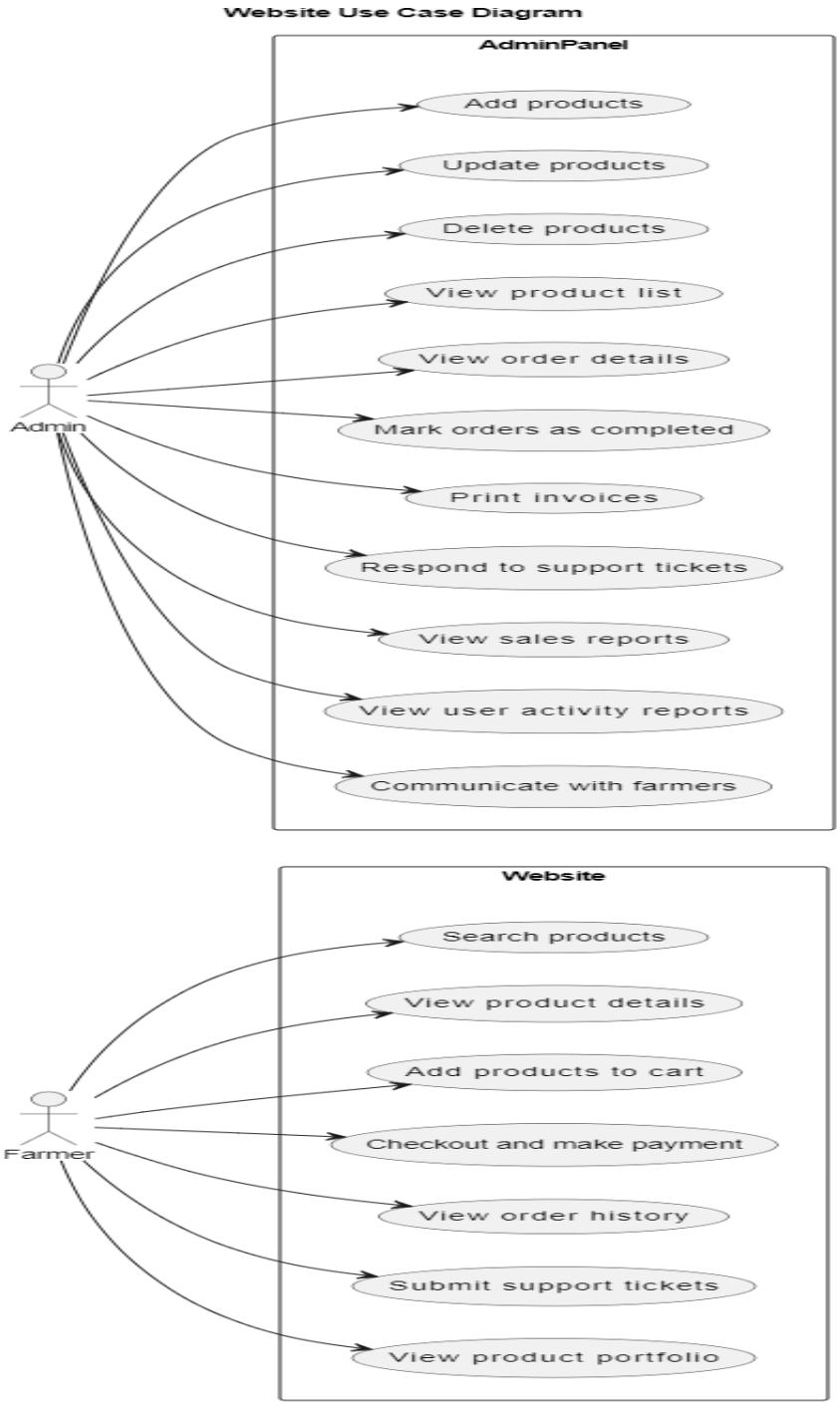


Figure 3: Use Case Diagram

8.2 Activity Diagram

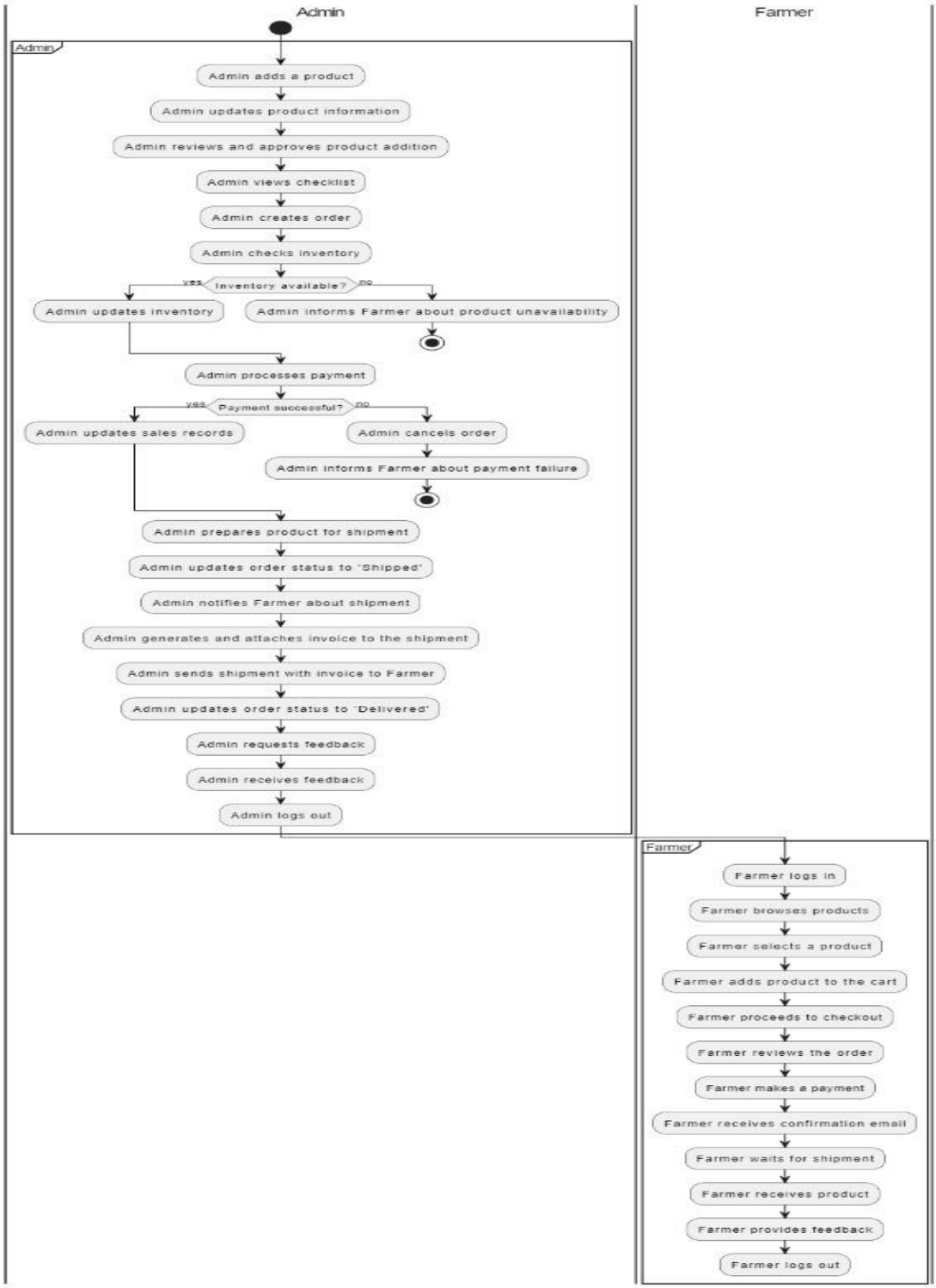


Figure 4: Activity Diagram

8.3 Requirement Catalogue

Functional Requirements:

- ✓ **FR1:** User registration and login features for farmers, agricultural professionals, and administrators.
- ✓ **FR2:** Ability for farmers to report and track crop-related concerns, illnesses, or emergencies.
- ✓ **FR3:** An agricultural case management system that tracks crop health, growth stages, and interventions.
- ✓ **FR4**: Integration with external agricultural service providers and specialists for collaboration and expert discussions.
- ✓ **FR5:** Crop advice process, providing individualized suggestions based on crop characteristics and farmer requirements.
- ✓ **FR6:** Communication tools that help farmers, agriculture specialists, and service providers communicate with one other.
- ✓ **FR7:** Able to provide analytics and reports on crop health, production rates, and productive farming methods.

Non-Functional Requirements:

- ✓ **NFR1**: An easy-to-use interface that is intuitive to use, even for farmers and agricultural specialists with different levels of technical proficiency.
- ✓ **NFR2**: Performance and scalability to accommodate a rising number of agricultural situations and users.
- ✓ **NFR3**: Security protocols to safeguard private information, such as user credentials and documents pertaining to crops.
- ✓ **NFR4:** To provide accessibility, compatibility with a variety of platforms and devices (web, mobile).
- ✓ **NFR5:** Quick reaction times and less downtime to guarantee that the system is ready when needed.
- ✓ **NFR6:** Adherence to pertinent ethical and legal standards pertaining to data privacy and agriculture.

User Interface Requirements:

- **UI1:** Clear and intuitive navigation for easy access to diverse features and capabilities.
- **UI2:** Responsive design that adjusts to various device and screen sizes.
- **UI3:** Visual cues and icons to help in understanding and utilizing the system.
- **UI4:** Farmers and agricultural specialists may customize their user profiles to control alerts and preferences.
- UI5: Support for many languages to appeal to a broad user base.

Performance Requirements:

- **PR1:** Fast loading speeds for agricultural data, service details, and other essential information.
- **PR2:** Effective search and filtering features for speedy discovery of particular
- **PR3:** dependable data synchronization and instantaneous updates for users and devices alike.

Security and Privacy Requirements:

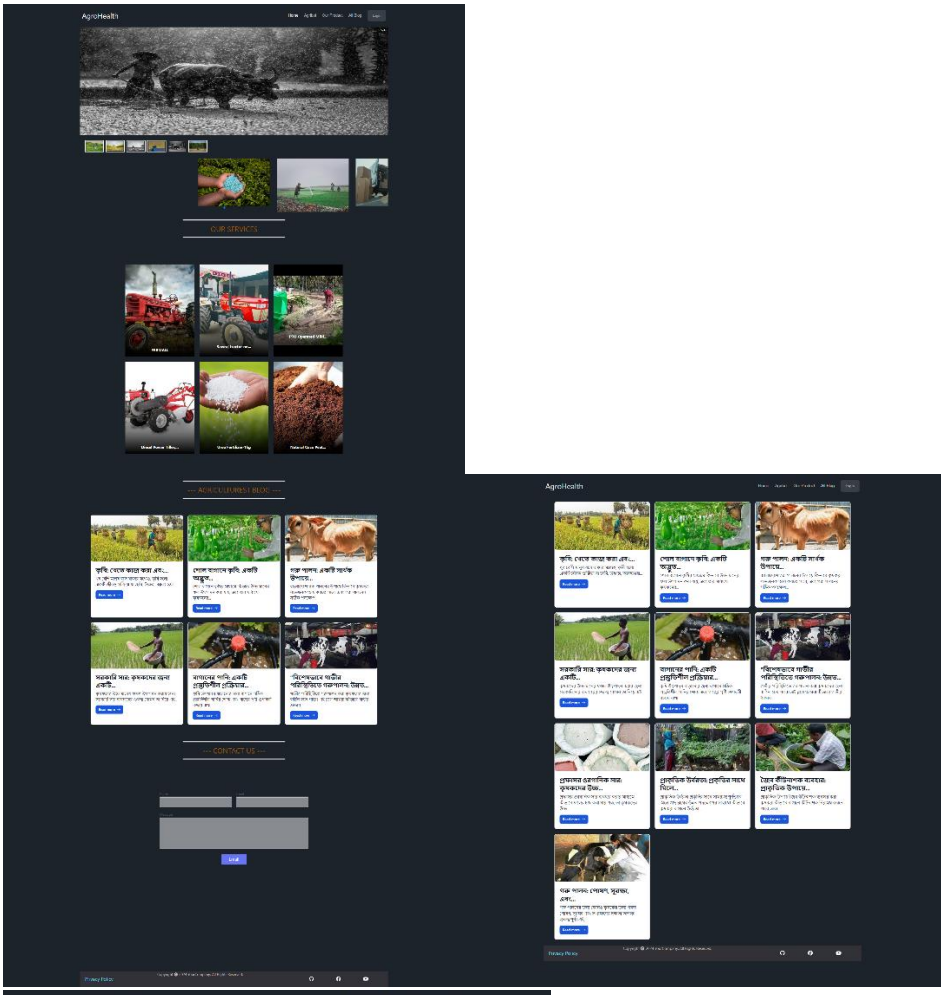
- **SR1:** Secure authentication and authorization systems to secure farmer accounts and data.
- **SR2**: Sensitive data encryption, including personal and crop-related data.
- **SR3:** Permission-based access control to guarantee that only authorized users may see and alter particular information.
- **SR4:** Compliance with data protection standards, safeguarding the confidentiality and privacy of farmer information.

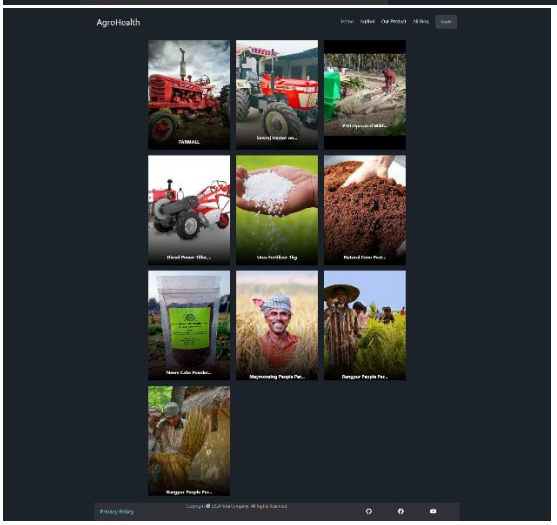
8.3 Prioritized Requirement List (PRL)

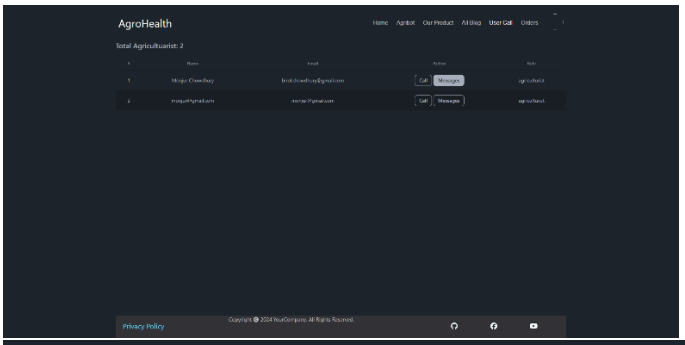
RequirementID	Requirement Description	Priority	Dependencies	Validation Criteria
AGH-RQ1	Functionality for user registration and login	High		Users are able to successfully register and log in.
AGH-RQ2	Document and monitor crop health problems	High		The ability to report and monitor health concerns
AGH-RQ3	A system for handling and documenting crop concerns	High		It is possible to create, assign, and amend issues.
AGH-RQ4	Cooperation with agronomic specialists and clinics	Medium	AGH-RQ3	smooth data transfer with outside systems
AGH-RQ5	Profile matching and crop recommendation procedure	Medium	AGH-RQ3	Appropriate pairing and guidance on agricultural matters
AGH-RQ6	Expert and farmer communication characteristics	Medium	AGH-RQ3	The system allows for efficient user communication
AGH-RQ7	Produce data and reports about the health of crops	Low	AGH-RQ3	The ability to provide accurate reports and data

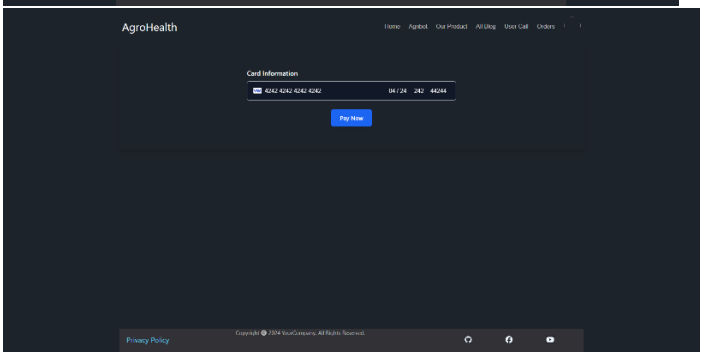
Table 5: Prioritized Requirement List

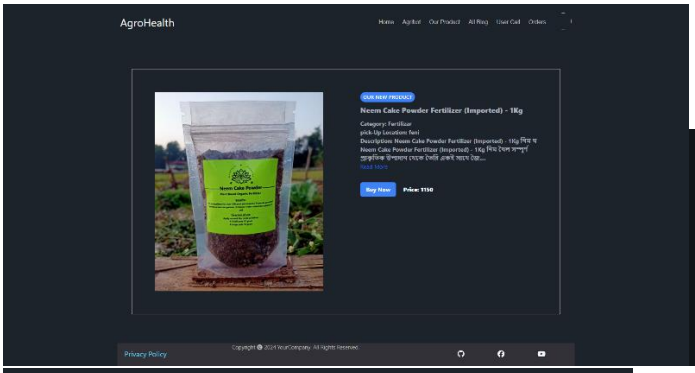
8.4 New system prototype

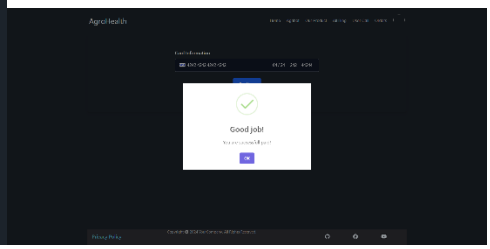


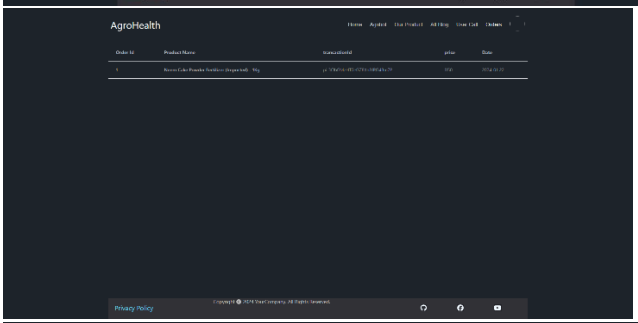


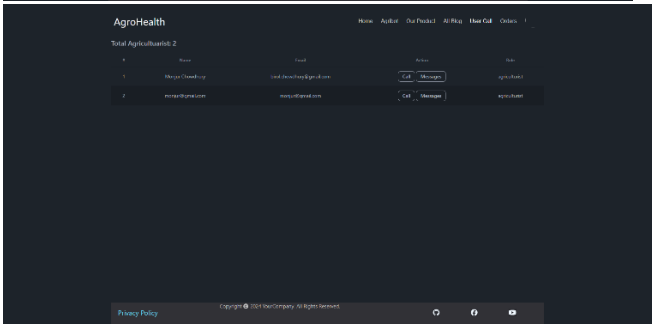

AgroHealth
Home Agribot Our Product All Blog User Call Orders
Total Agriculturalist: 2
Privacy Policy


AgroHealth
Home Agribot Our Product All Blog User Call Orders
Card Information
Pay Now
Privacy Policy


AgroHealth
Home Agribot Our Product All Blog User Call Orders
Neem Cake Powder Fertilizer (Imported) - 1Kg
Buy Now
Price: 1150
Privacy Policy


AgroHealth
Good job!
OK


AgroHealth
Home Agribot Our Product All Blog User Call Orders
Privacy Policy


AgroHealth
Home Agribot Our Product All Blog User Call Orders
Total Agriculturalist: 2
Privacy Policy

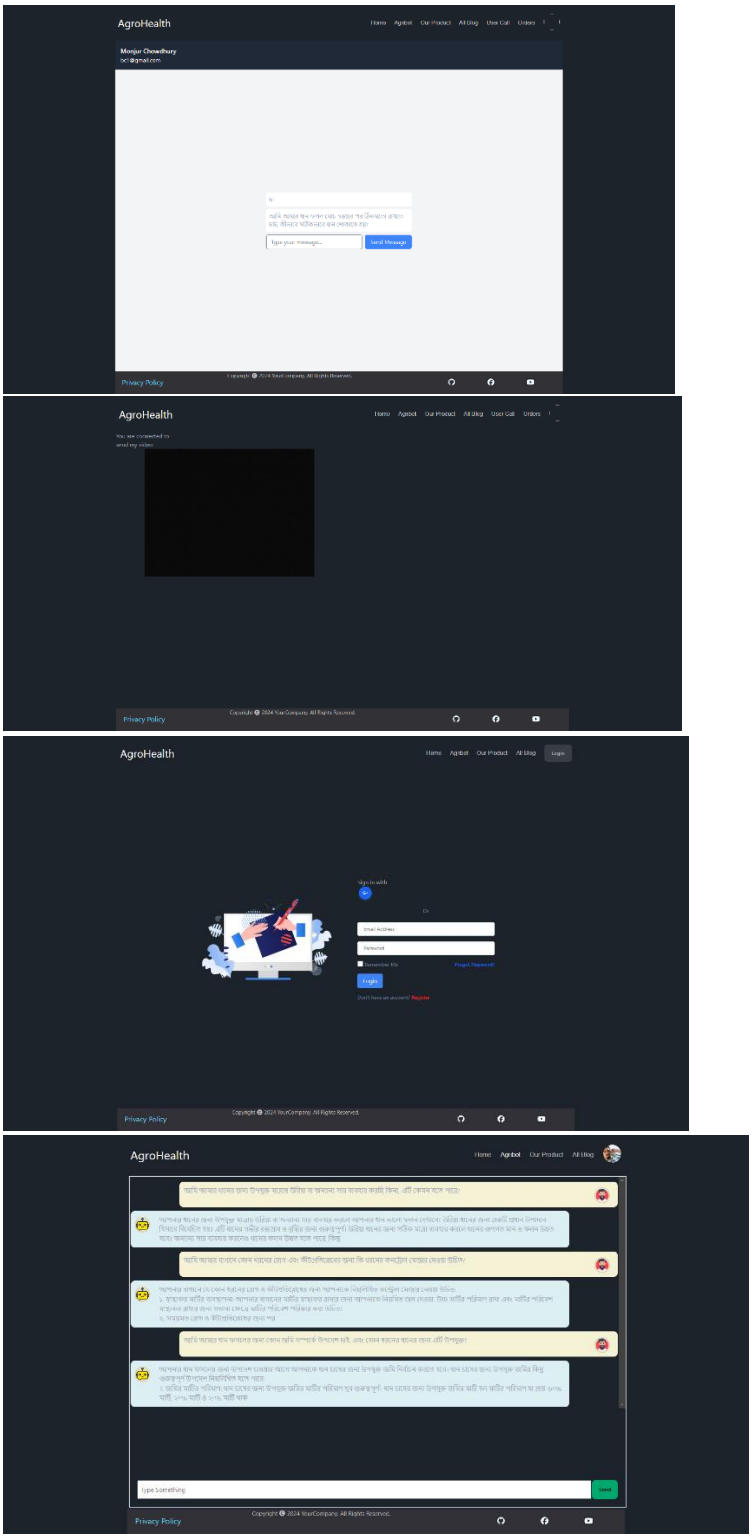


Figure 5: Prototype of new system

CHAPTER 9: ENGINEERING

The AgroHealth web app has a plethora of additional modules because it is a customizable feature. s a result, there will be different modules to clarify. I'll go over some of the essential. Here are the modules:

- **User Registration and Profiles:** Farmers have the ability to register as users and maintain their profiles, which include contact data, and agricultural preferences.
- **Crop Management:** Farmers may enter and maintain information about their crops in this module, such as the type of crop, when to plant, when to harvest, and any applicable agricultural techniques.
- **Service Requests**: Farmer requests for assistance with pest management, soil testing, irrigation support, and other farming-related services can be submitted.
- **Expert Consultation:** Farmers can consult with agricultural specialists using this module. On crop management, disease control, and other farming techniques, farmers may make appointments for consultations, ask questions, and get assistance.
- **Marketplace:** Enabling farmers to post their production, peruse available goods, and conduct business with buyers or sellers, this module facilitates the buying and selling of agricultural items.
- **Educational Resources:** This module provides articles, books, and other educational resources on cutting-edge agricultural advancements, sustainable farming methods, and modern farming techniques.
- **Sharing of Equipment:** Farmers can exchange information on the tools, machinery, and agricultural equipment that is available. This makes it easier for the agricultural community to work together and share resources.

9.1 Class Diagram

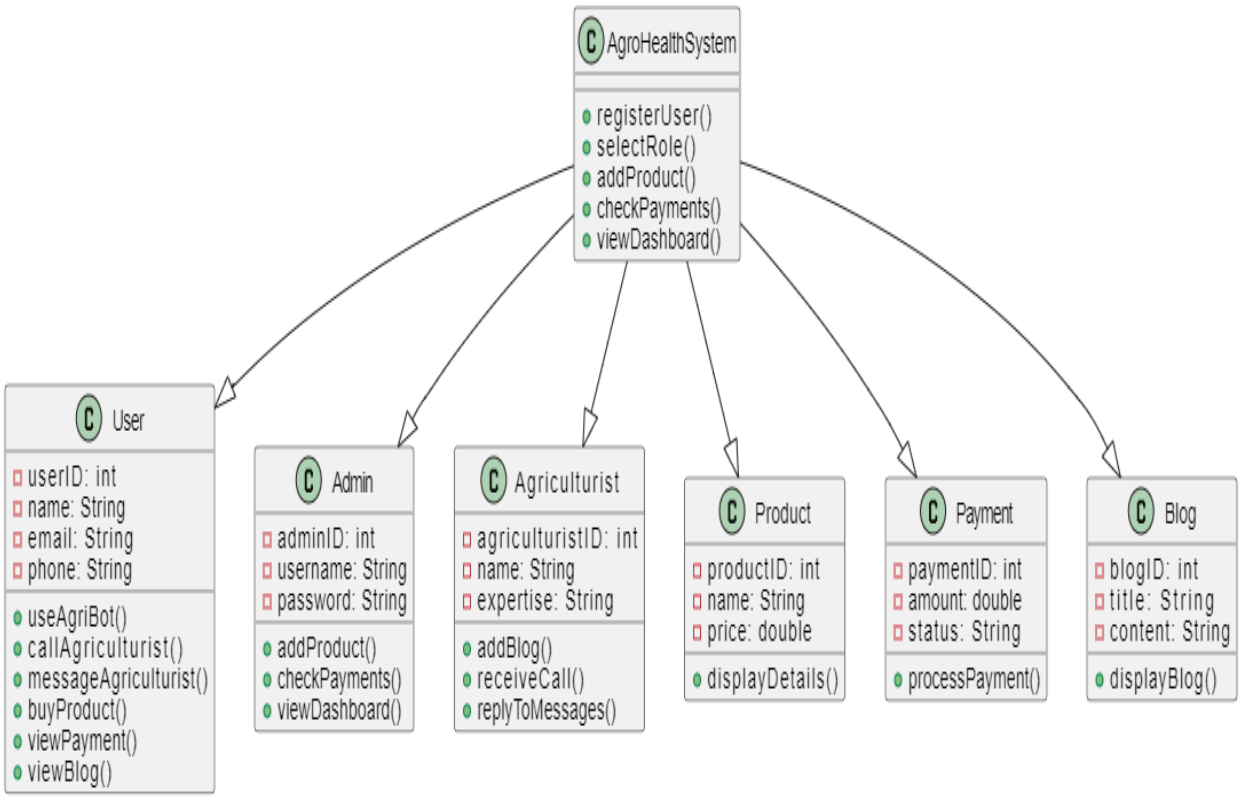


Figure 6: Class Diagram

9.2 Use Case Diagram

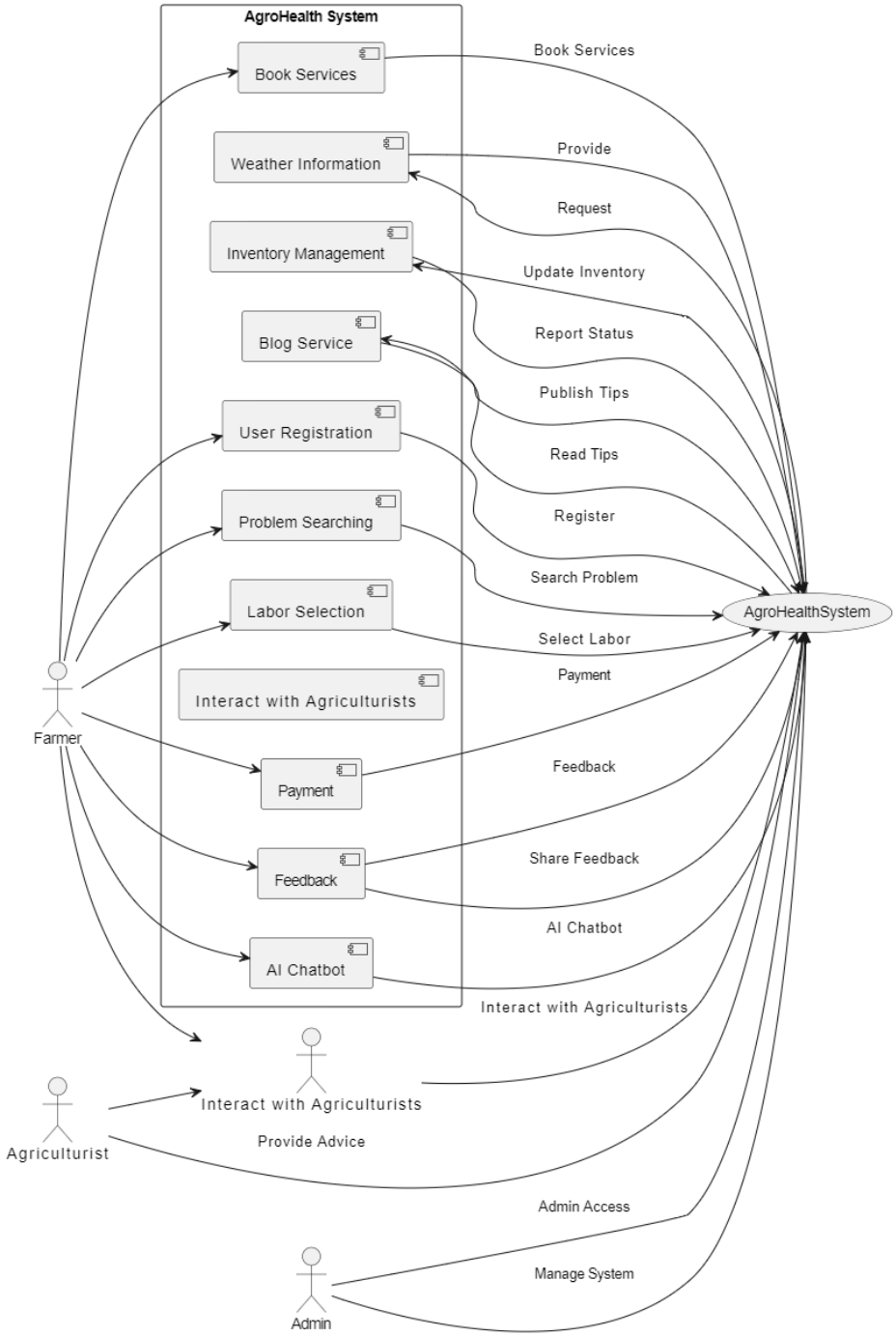


Figure 7: Use Case Diagram

CHAPTER 10: Development

10.1 Samples of Core Module Coding

```
app.use(cors())
app.use(bodyParser.json())

const emailtosocketMapping = new Map()
const socketToEmailMapping = new Map()

io.on("connection", (socket) => {
  socket.on('join-room', data => {
    const { roomId, emailId } = data
    console.log("user", emailId, "joinRoom", roomId);
    emailtosocketMapping.set(emailId, socket.id)
    socketToEmailMapping.set(socket.id, emailId)
    socket.join(roomId)
    socket.emit("joined-room", { roomId })
    socket.broadcast.to(roomId).emit("user-joined", { emailId })
  });
  socket.on('call-user', (data) => {
    const { emailId, offer } = data
    const fromEmail = socketToEmailMapping.get(socket.id)
    const socketId = emailtosocketMapping.get(emailId)
    console.log("call user", socketId);
    socket.to(socketId).emit("incomming-call", { from: fromEmail, off
  });
  socket.on('call-accepted', (data) => {
    const { emailId, ans } = data;
    console.log("call accect", emailId);
    const socketId = emailtosocketMapping.get(emailId)
    console.log(socketId);
    socket.to(socketId).emit('call-accepted', { ans })
```

```
app.get('/myorderlist', async (req, res) => {···
})

// save DB Payment
app.post("/payments", async (req, res) => {
  const payment = req.body;
  const paymentProduct = await paymentCollecton.insertOne(payment
  res.send(paymentProduct)

})
app.get("/allpayments", async (req, res) => {
  const result = await paymentCollecton.find().toArray();
  res.send(result)

})
// add a blog
app.post("/addblogs", async (req, res) => {
  const added = req.body;
  const result = await blogCollection.insertOne(added);
  res.send(result)
})
// get all blog
app.get("/addblogs", async (req, res) => {
  const result = await blogCollection.find().toArray();
  res.send(result)
})
```

```
async function run() {
  try {
    // Connect the client to the server (optional starting in v4.7)
    // await client.connect();


    const userCollecton = client.db('Agrohealth').collection('users');
    const productCollection = client.db('Agrohealth').collection('productList');
    const blogCollection = client.db('Agrohealth').collection('blog');
    const paymentCollecton = client.db('Agrohealth').collection('payment');
    const messageCollecton = client.db('Agrohealth').collection('message');

    //user get
    app.get("/userslist", async (req, res) => {
      const result = await userCollecton.find().toArray();
      res.send(result)
    })
    app.get("/user", async (req, res) => {···
    })
    // user save in DB
    app.post('/users', async (req, res) => {···
    })
    // user role set in DB
    app.patch('/users/role/:id', async (req, res) => {···
    })

    // getAll product
    app.get("/allproduct", async (req, res) => {···
```

```
// OpenAi chatBot using with OpenAi
app.post("/chat", (req, res) => {
  const question = req.body.question;
  openai.completions.create({
    model: "gpt-3.5-turbo-instruct",
    prompt: question,
    max_tokens: 400,
    temperature: 0,
  }).then((response) => {
    return (response?.choices?.[0]?.text);
  })
    .then((answer) => {
      const array = answer?.split("\n").filter((value) => value).map((value) => value.trim())
      return array
    }).then((answer) => {
      res.json(
        {
          answer: answer,
          Prompt: question
        }
      )

    });

})

}
finally {
```

```
run().catch(console.dir);
app.get('/', (req, res) => {
  res.send('Hello Server')
})

io.listen(8001)
app.listen(port, () => {
  console.log(`Example app listening on port ${port}`)
})
```

Figure 8: Code sample

10.2 Possible problem break down

- ✓ **Compatibility Problems:** Various iterations of Express.js, MongoDB, and React may present compatibility issues with particular dependencies or libraries. Potential conflicts can be reduced by maintaining compatibility and upgrading these technologies on a regular basis.
- ✓ **Performance Challenges:** MongoDB's heavy database operations and React frontend's handling of massive datasets or intricate interactions may cause performance problems. These issues may be resolved by putting effective algorithms into practice, query optimization, and the application of React's speed optimization strategies.
- ✓ **Platform-Specific Restrictions**: React facilitates cross-platform development; nonetheless, certain subtleties or restrictions may be unique to

a certain platform. A flawless application may depend on guaranteeing a consistent user experience and resolving any platform-specific problems.

- ✓ **State Management Complexity:** Careful thought is needed to ensure efficient state management with React, particularly for intricate application states. State-related issues may be resolved by implementing suitable state management patterns or tools, such Redux or Context API, and skillfully organizing the application architecture.
- ✓ **Restricted Package Support**: Although the npm package ecosystem is large, some functionality or integrations could not have easily accessible packages. In certain situations, custom implementations or contributions to open-source projects could be required.
- ✓ **Consistency of User Interface:** It is crucial to provide a uniform and adaptable user interface across several devices and browsers. Maintaining UI uniformity may be achieved by following responsive design guidelines and carrying out extensive testing across various hardware and browser combinations.
- ✓ **Data Synchronization and Offline Support: React** offline support, handling data conflicts, and implementing strong data synchronization in the Express.js backend can be difficult to do. These difficulties can be lessened by putting caching systems in place, addressing offline circumstances, and prioritizing data management solutions.
- ✓ **Security and Data Protection:** It is important to take precautions to secure communication in the Express.js backend and to protect user data in MongoDB. Data security may be improved by using strong authentication procedures, encryption methods, and secure coding standards.
- ✓ **Testing and Debugging:** It might take a while to find and fix problems, such as inconsistent behavior or backend/frontend flaws. Prioritizing rigorous testing, employing debugging tools, and following best practices for automated testing in both Express.js and React will help detect and fix these problems efficiently.
- ✓ **Limited Resources and Community** Support: In situations where there is little documentation or community support, it may be difficult to get resources for troubleshooting or solving certain problems. Getting involved in the React, Express.js, and MongoDB communities, asking for advice, and contributing to discussions can assist get around these restrictions.

10.3 Prioritization while developing

Prioritization	Requirements and Explanation
1. Core Functionality	Lookup and guidance on Farmer their agricultural field reporting problems with crop health Making contact with clinics or specialists in agriculture Give top priority to putting in place aspects that are necessary for managing crop health, such as expert networking, advice retrieval, and effective issue reporting. Make sure these features are easy to use and cohesively included.
2.User Experience (UX)	a tidy and agile user interface Simple to use vivid information display Prioritize delivering a smooth and understandable user experience. Test the usability with users and collect feedback to make usability better over time.
3.Data Management and Security	Strong data encryption safe authentication of users Controls over access To establish and preserve user trust, make sure that user data is managed securely and give top priority to data encryption, secure authentication, and access controls.
4. Performance Optimization	Code optimization for performance improvement reducing the number of network queries Effective data caching Reduce the number of network queries made, concentrate on writing efficient code, and use efficient data caching to speed up loading times to maximize the performance of your application.
5. Integration with External Systems	Connectivity with farming databases Integration with professional scheduling platforms Prioritize system integration wherever applicable to improve the functionality of the app and provide consumers access to a wealth of data and services.

6. Emergency and Urgent Assistance	Prompt notification of critical crop health problems Quick communication with emergency personnel If the app deals with emergency issues, given top priority to features that let users report emergency instances, get in contact with emergency services, or get help right away.
7. Testing and Quality Assurance	Automated Testing units Testing for user acceptability Prioritize extensive testing and quality control to guarantee a reliable and error-free application. Give user acceptability and automation testing top priority in order to identify and address problems before release.
8. Continuous Improvement andUser Feedback	Compiling user input Regular updates are released Make it a priority to actively collect user input and implement recommendations into the app's development. Release updates often to meet user requests, correct errors, and add new features in response to feedback from users.

Table 6: Prioritization, Requirement & Explanation

CHAPTER 11: TESTING

Project Name	AgroHealth Web application		
Name of product	AgroHealth app		
Product description	AgroHealth app		
Project description	MERN stack web application		
Project duration	Project Type	Testing/ Verification	
	Start date	End date	
	16 Sep 2023	25 Sep2023	

11.1 Acceptance of the Test Plan

Specify goals ,parameters of the testcase. Determine the important parties and get their consent to the test plan. Clearly define each testing phase's acceptance criteria.

Name of test Case	Unit test		
Test Class	Admin Panel		
Test Description	The Admin Panel class tests the functionality of the admin panel in the Agro-Health Web App.It includes test methods that cover actions performed by the admin, such as logging in, managing Product listings, handling user requests, and managing different services. These tests make sure the admin panel performs as intended and exhibits the desired behavior.		
Source of data	Test Steps	ProceduresActual Result	Expected Result
Admin	Open admin panel login page. Enter valid credentials (username and password). Click on the "Login" button.	Admin shouldbe successfully logged in and redirected to the admin panel home page.	The admin is successfully logged in and redirected to the admin panel home page.

Name of test Case	Unit		
Test Class	Admin Panel		
Test Description	The Admin Panel class tests the functionality of the admin panel in the AgroHealth Web App.This specific test case focuses on the scenario of viewing the Product listings in the admin panel.		
Source of data	Test Steps	ProceduresActual Result	Expected Result

Admin	Login to the admin panel. Navigate to the "Product Listings" section.	The admin should be able to view the list of products with their relevant details.	The actual result depends on the execution of the test case and may vary.

11.2 Test Case

Create an extensive collection of test cases that encompass every feature and functionality of the program. Incorporate border cases, edge situations, and positive and negative test scenarios. Make sure test cases are thoroughly documented, with procedures, expected outcomes, and preconditions included.

Test Case Name	Unit/ Module /Integration test		
Test Class			
Test Description			
Source of data	Test Steps	Expected Result	Actual Result

11.3 Unit Testing (2 to 3)

Test each program module or component separately using unit testing. Create test cases that are tailored to each unit's functionality. Make sure every unit operates independently and appropriately.

Table 7: Unit Testing

11.4 Module Testing (2 to 3)

Test the interaction between many units/modules. Verify that modules are able to interact and communicate with one other. Check the integrity and proper flow of data between modules.

Name of test Case	Module		
Test Class	Admin Module		
Test Description	The Admin Module class is responsible for testing the module related to the admin functionality in the AgroHealth App. It includes test methods that cover specific scenarios and actions performed by the admin user.		
Source	Test Steps	ProceduresActual Result	Expected Result
Admin	Initialize admin module. Call the login() method with valid credentials. Check the returned value or status of the login operation.	It should be possible for the admin to access the admin panel and log in.	The actual result depends on the execution of the test case and may vary.

Name of test Case	Module
Test Class	Admin Module
Test Des.	

Source of data	Test Steps	ProceduresActual Result	Expected Result
Admin	Initialize the admin module. Call the getProductListings( )method. Retrieve the list of Product listings fromthe section. Examine the differences between the predicted and retrieved product listings.	The admin should be able to retrieve a list of Product listings with their relevant details.	The actual result depends on the execution of the test case and may vary.

Table 7: Module Testing

11.5 Integration Testing (2 to 3)

Verify that each module is integrated and compatible with the others. Determine and fix any problems with dependencies, data exchange, or communication. Check to see if the integrated system functions as planned.

Name of Test Case	Integration		
Test Class	Admin Integration		
Test Description	Testing the integration of the admin features in the AgroHealth App is the responsibility of the Admin Integration Test class. It focuses on how various parts or modules work together to guarantee the admin functions operate without a hitch.		
Data source	Test Steps	ProceduresActual Result	Actual Result

Admin	Initialize the admin module. Simulate a user login by calling the login() method with valid credentials. Check the returned value or status of the login operation. Verify that the user is redirected to the admin panel home page.	The admin should be able to successfully log in and access the admin panel.	The actual result depends on the execution of the test case and may vary.

Table 8: Integration Testing

11.6 Acceptance Testing

To make sure the application satisfies the criteria, do acceptance testing. Examine every feature of the program from the viewpoint of the user. Involve stakeholders and get their opinions on the functionality and usability of the program.

Test Case Name	Acceptance test		
Test Class			
Test Description	The Admin Acceptance Test class is responsible for conducting acceptance testing for the admin functionality inthe AgroHealth App. It focuses on ensuring that the admin features meet the specified requirements and user expectations.		
Data source	Test Steps	ProceduresActual Result	Expected Result

Test Case(Login Functionality)	Open the admin panel login page. Enter valid credentials (username and password). Click on the "Login" button. Observe the behavior of the system. Expected Result:	The admin should be able to successfully log in and be redirected to the admin panel home page.	The actual result depends on the execution of the test case and may vary.

Table 9: Acceptance Testing

11.7 Usability Testing

Analyze how intuitive and user-friendly the program is. Get opinions on the navigation, apps, and general user experience. Determine what needs to be improved upon and make the required adjustments.

Test Case Name	Admin Usability Test	
Test Class		
Test Description	The Admin Usability Test class is responsible for conducting usability testing for the admin functionality in the AgroHealth web-App. It focuses on evaluating the user interface,user experience, and ease of use of the admin panel.	

Data source	Test Steps	ProceduresActual Result	Actual Result
	Open admin panel login page. Attempt to find the login form easily. Enter valid credentials (username and password).Click on the "Login" button. Observe the user interface and overall usability during the login process.	The login form should be prominently displayed and easy to locate. The login process should be intuitive and straightforward.	The actual result depends on the execution of the test case and may vary.

Table 10: Usability Testing

CHAPTER 12: INPLEMENTATION

12.1 Training

User	Training	Comment
Admin		
Agriculturist		

12.2 Big Bang Implementation

The AgroHealth project's Big Bang implementation strategy calls for the simultaneous rollout of the complete system, including the Express.js-built backend, MongoDB for the database, and the React frontend, without the need for a pilot or parallel deployment. A thorough training program will be necessary for all users, including farmers, agriculturalists, and administrators, in order to ensure a seamless changeover. The training sessions will address a range of topics, including interacting with farmers, using agricultural services, navigating the site, and any other relevant features and functionality. Ensuring that every user is fully conversant with the AgroHealth platform is crucial in order to enable them to effectively employ its many functionalities for their farming requirements.

12.3 Scaling

The AgroHealth project requires scaling, which entails getting the software ready to handle more users, more data, and more transactions. The following lists particular training factors to be aware of when scaling:

12.3.1 Scalable Design

Make sure the development and architecture staff are trained in building an effective and scalable AgroHealth system. Comprehending ideas like as database optimization, caching tactics, horizontal scaling approaches, and efficient performance monitoring implementation are necessary for this.

12.3.2 Performance Testing

Conduct performance testing to determine AgroHealth's capabilities to manage growing demands. Teach the testing team to spot performance problems and bottlenecks during testing so that the software may be optimized and its features increased.

12.3.3 Monitoring and Optimization

Give the operations staff training so they can keep an eye on AgroHealth's performance and make the most use of its resources. This includes keeping an eye on the health of the server, rating database performance, analyzing network traffic, and monitoring other crucial metrics to make sure the application runs smoothly and effectively.

12.4 Load Balancing

Load balancing is essential for maximizing performance and resource use in the AgroHealth project. The following goes over load balancing training considerations:

12.4.1 Load Balancer Configuration

Make that the operations staff has received training on load balancer configuration and management for AgroHealth. This entails knowing how to set up health checks, distribute traffic effectively, and comprehend load balancing algorithms.

12.4.2 Handling Scalability Events

Teach the support and operations staff how to effectively manage AgroHealth-specific scaling events. This involves managing any problems quickly, guaranteeing continuous service continuity, and adding or deleting servers from the load balancer pool.

12.4.3 Monitoring and Optimization

Train the operations staff on how to keep an eye on the load balancer's efficiency and customize its settings for AgroHealth. To ensure efficient load balancing, this entails close observation of traffic distribution, response times, and server health.

CHAPTER 13: CRITIVAL APPRAISAL & ECALUATION

13.1 Potentially achievable goals

- ✓ Increasing the effective settlement of farming difficulties and improving the reaction time for agricultural crises are possible goals for the AgroHealth initiative.
- ✓ Offer farmers fast and accurate agricultural advice.
- ✓ Enable effective communication among volunteers, agricultural specialists, and farmers.
- ✓ Keep an extensive record of agricultural cases for monitoring and aid with follow-up.
- ✓ Promote possibilities for agricultural support and sustainable farming techniques.

13.1.1 Percentage of success for each goal

The resolution of agricultural issues, the speed at which farmer inquiries are answered, the effective application of agricultural guidance, the effectiveness of communication, the accuracy of the database, and the involvement of users in educational campaigns are all indicators of the success rate of each AgroHealth project objective.

13.1.2 How much more excellent work was possible?

There is need for improvement in terms of quicker response times, more effective issue resolutions, better agricultural advising methods, better communication and user interface tools, more database capacity, and reaching a larger audience for educational programs.

13.1.3 The reasons it was not possible

Certain goals may not be met because of issues like few funds, resources, or access to sophisticated agricultural knowledge in far-off places. Furthermore, factors might include lack of knowledge among potential consumers or technological difficulties.

13.1.4 Which goals have been abandoned?

For the reasons outlined above, the application may have fallen short of its goals regarding the effective handling of farming-related problems, database accuracy, and the outreach of educational initiatives.

13.1.5 The reasons these goals have not been met

These goals might not have been met because of logistical difficulties, a lack of agricultural specialists in some areas, poor user involvement, or a lack of commercial support for instructional programs.

13.1.6 How those goals may have been accomplished

More funds may have been requested to upgrade infrastructure and provide access to agricultural knowledge in order to fulfill those goals. To boost user involvement and expand the audience for educational activities, targeted marketing and outreach campaigns may have been carried out.

13.1.7 How much better are the solution's features?

One may argue that the AgroHealth application's characteristics are beneficial for improving the efficiency and coordination of dealing with agricultural difficulties. The software may speed up stakeholder communication and offer useful agricultural data, which would ultimately enhance farming methods and problem-solving techniques.

13.1.8 Which characteristics were immobile

It's possible that certain features weren't completed because of time or money restrictions. Advanced AI-based diagnostic tools, agricultural support teams' real-time tracking, and interaction with regional agricultural institutions and governmental organizations are a few examples of these aspects.

13.1.9 The reason these attributes couldn't be altered

The team may not have been able to completely integrate these sophisticated features due to a lack of funding and development time.

13.1.10 How may such characteristics be addressed?

The AgroHealth project might receive more money and a specialized development team in order to incorporate the undeveloped elements. Working together with universities or organizations dedicated to agriculture may also offer the know-how needed to include cutting-edge features into the program.

13.2 Unfulfilled Goals

Some AgroHealth project goals, like reaching particular crop yield improvements targets or extending agricultural support to a larger geographic area, might not have been accomplished because of obstacles like scarce funding, insufficient access to cutting-edge technologies, and challenges in reaching outlying farming communities. Unpredictable weather patterns and other external circumstances are another reason why these objectives are not achieved. Resolving these issues is

essential for making plans for the future and getting beyond obstacles in the way of achieving particular goals.

13.2.1 Causes of Unfulfilled Goals

These unmet goals in the AgroHealth project might be the consequence of poor user uptake, lack of requisite infrastructure, logistical difficulties, or budgetary limitations.

13.2.2 Possible Remedies

The project team may have worked with governmental or non-profit groups to pool resources and expertise, obtained more money, and carried out thorough feasibility studies to solve these unfulfilled goals. Regular testing, debugging, and user input should have been essential components of software development. Extensive documentation, encompassing user manuals and technical instructions, would facilitate comprehension and subsequent upkeep.

CHAPTER 14: LESSONS LEARNED

14.1 Prior Project

Establishing the goals, parameters, and scope of the project are crucial at the pre-project stage of the AgroHealth application. Researching the market, determining the target audience, and outlining features are all necessary for this. Important actions include assembling the staff and technology needed, as well as drafting a thorough project plan with deadlines and milestones.

14.2 Examine

Periodic reviews are required during the project to evaluate the status, spot deviations from the plan, and guarantee that the objectives are being met. Feedback from stakeholders, users, and the team is essential for assessing the app's efficacy and making any necessary modifications or enhancements.

14.3 The End

In the closure phase, the AgroHealth project is concluded, unfinished business is resolved, lessons learned are recorded, final testing is carried out, and deployment preparations are made. A post-project review evaluates the project's overall performance and collects input for future

14.4 Acquired Knowledge

It's critical to consider both professional and personal development. Gaining insight from the AgroHealth project's lessons, difficulties encountered, and skills obtained helps improve one's capacity for future undertakings.

14.5 Issues Raised

The AgroHealth initiative may face obstacles in the form of communication breakdowns, technological problems, resource shortages, or challenges with data gathering. Efficient problem-solving requires a thorough documentation of these problems and their causes.

14.6 Resolutions Took Place

When issues emerge, practical fixes are crucial. Solutions for the AgroHealth initiative include working with neighborhood organizations, putting data validation into practice, resolving technological problems, setting priorities for resources, and encouraging open lines of communicate.

CHAPTER 15: CONCLUSION

15.1 Project Synopsis

The goal of the AgroHealth project was to provide a complete approach to agricultural well-being. In order to improve overall farm health, the application sought to link farmers, and agricultural specialists. Through the use of technology, the initiative improved agricultural well-being, pest control, and crop health procedures.

15.2 The Project's Objective

The main objective was to create a productive and intuitive program that would enable farmers, medical professionals, and agricultural specialists to work together more easily. The application sought to enhance overall agricultural well-being, pest management, and crop health. The objective was to improve agricultural well-being, boost output, and promote community cooperation.

15.3 The Project's Success

Key performance indicators might be used to gauge the project's success:

- ✓ Better pest control and crop health.
- ✓ Better cooperation and communication between farmers and agricultural specialists.
- ✓ Positive comments and pleasure from users.
- ✓ Adoption of the application among the agricultural community.
- ✓ Positive influence on overall farm production and well-being.

15.4 Records

Most likely, the documentation addressed:

- ✓ **Pre-Project Documentation:** Project ideas, feasibility studies, and early requirements collection.
- ✓ **Project Plan:** Objectives, scope, timeframes, resource needs, and risk management.

- ✓ **Technical Specifications:** Detailed documentation explaining the application's architecture, features, and functionalities.
- ✓ **User Documentation:** Guides for farmers, health professionals, and agricultural specialists on properly utilizing the program.
- ✓ **Testing and Documentation for Quality Assurance:** Test strategies, scenarios, and outcomes guaranteeing the caliber of applications.
- ✓ **Application Deployment and Maintenance Plans:** Procedures for

15.5 Project Value

The potential for the AgroHealth initiative to have a major influence on agricultural well-being is what makes it valuable. The application helps with farm production, pest control, and general agricultural performance by promoting teamwork and offering focused health solutions. By raising awareness and fostering a community of agricultural stakeholders, the initiative also generates value.

15.6 My Personal Experience

My experience with the AgroHealth project included managing schedules, liaising with stakeholders, addressing technical issues, and putting customized solutions into place. I have held positions such as project manager and developer. Through the development of my problem-solving, teamwork, and documentation skills, I was able to further my career in the field of farm technology

Plagiarism Report

201-16-501

ORIGINALITY REPORT

7%	6%	0%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	3%
2	dspace.daffodilvarsity.edu.bd:8080 Internet Source	2%
3	Submitted to University of Greenwich Student Paper	1%
4	fastercapital.com Internet Source	<1%
5	Submitted to Visvesvaraya Technological University Student Paper	<1%
6	Submitted to Wellington Institute of Technology Student Paper	<1%
7	Submitted to Aliat Universidades Student Paper	<1%
8	Submitted to Universiti Putra Malaysia Student Paper	<1%
9	Submitted to Victoria University	