# AGRISENSE: FARMER’S ASSISTANT

**KUTO VICTOR**

**A research project submitted to the School of Science, Engineering and Technology in partial fulfillment of the requirements for the award of Diploma in Information Technology, Kabarak University.**

**NOVEMBER,2025**

# COPYRIGHT

©2025

All rights reserved

No part of this project may be reproduced, stored in any retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, and recording without prior intention of the author or Kabarak University on that.

Sign: \_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_

# DECLARATION AND APPROVAL

I declare that this work without any reasonable doubt has never been presented before to the Faculty of Information Technology or any other institution. No part of this research document shall therefore be duplicated without prior consent.

**Sign**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Date**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student: Kuto Victor Kiprotich

Registration Number: DIT/N/0868/05/24

# RECOMMENDATION

This research project entitled Agrisense: Farmers Assistantwritten by Kuto Victor Kiprotich is presented to the School of Science, Engineering and Technology, Kabarak University. I have reviewed this project and recommend that it will be accepted in partial fulfillment of the requirements for the Diploma in Information Technology.

**Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_ **Date**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Mr. Elvine Saikwa**

**School of Science, Engineering and Technology**

**Department of Computer Science and Information Technology**

**Kabarak University**

# ACKNOWLEDGMENT

I sincerely thank the Almighty God for strength and guidance throughout this project. My heartfelt appreciation goes to my supervisor for their valuable support, advice and encouragement during the period of this work. I also extend my gratitude to my lecturers, classmates and family for their continuous assistance, support and motivation which greatly contributed to the completion of this project.

# DEDICATION

This project is dedicated to my loving family for their strong support, encouragement and prayers. I also dedicate it to my friends and lecturers who have continuously inspired me to work hard towards my goal in my life.

# ABSTRACT

Agrisense is an intelligent, AI- powered agricultural platform designed for plant identification, disease diagnosis, weather-based farming insights and community knowledge sharing. Its goal is to empower farmers with accessible, data-driven tools for modern agriculture, reduce reliance on manual process, improve productivity and contribute to sustainable farming.

**Keywords**: *Agrisense, AI, Farm analytics*

TABLE OF CONTENT

[AGRISENSE: FARMER’S ASSISTANT](#_Toc214829741)

[COPYRIGHT ii](#_Toc214829742)

[DECLARATION AND APPROVAL iii](#_Toc214829743)

[RECOMMENDATION iv](#_Toc214829744)

[ACKNOWLEDGMENT v](#_Toc214829745)

[DEDICATION vi](#_Toc214829746)

[ABSTRACT vii](#_Toc214829747)

[LIST OF FIGURES xi](#_Toc214829748)

[LIST OF TABLES xii](#_Toc214829749)

[CHAPTER ONE 1](#_Toc214829750)

[INTRODUCTION 1](#_Toc214829751)

[1.1 Background 1](#_Toc214829752)

[1.2 Problem Statement 1](#_Toc214829753)

[1.3 Purpose of the study 2](#_Toc214829754)

[1.4 Objectives of the study 2](#_Toc214829755)

[1.5 Research Questions 2](#_Toc214829756)

[1.6 Justification of the study 2](#_Toc214829757)

[1.7 Scope of the study 2](#_Toc214829758)

[1.8 Study Limitations 3](#_Toc214829759)

[1.9 Assumptions of the Study 3](#_Toc214829760)

[CHAPTER TWO 4](#_Toc214829761)

[LITERATURE REVIEW 4](#_Toc214829762)

[2.0 Introduction 4](#_Toc214829763)

[2.1 Digital Agriculture and Technology Adoption 4](#_Toc214829764)

[2.2 System Functionality 5](#_Toc214829765)

[2.2.1 Plant Identification and AI-Powered Systems 5](#_Toc214829766)

[2.2.2 Disease Diagnosis Systems and Challenges 5](#_Toc214829767)

[2.2.3 Weather Integration and Farm Decision-Making 6](#_Toc214829768)

[2.3 Community Knowledge-Sharing Platforms 6](#_Toc214829769)

[2.4 Related Work and Existing Solutions 6](#_Toc214829770)

[2.5 Gaps in Existing Solutions 6](#_Toc214829771)

[2.6ConceptualFramework 7](#_Toc214829772)

[CHAPTER THREE 8](#_Toc214829773)

[RESEARCH METHODOLOGY 8](#_Toc214829774)

[3.0 Introduction 8](#_Toc214829775)

[3.1 System Development Methodology 8](#_Toc214829776)

[3.2 Requirements Analysis 9](#_Toc214829777)

[3.2.1 Functional Requirements 9](#_Toc214829778)

[3.2.2 Non-Functional Requirements 9](#_Toc214829779)

[3.3 System Development Tools and Technologies 10](#_Toc214829780)

[3.4 Data Collection Methods 11](#_Toc214829781)

[3.4.1 Primary Data Collection 11](#_Toc214829782)

[3.4.2 Secondary Data Collection 11](#_Toc214829783)

[3.5 System Architecture Overview 12](#_Toc214829784)

[3.6 Data Flow Diagram 12](#_Toc214829785)

[3.6.1 Context Diagram 12](#_Toc214829786)

[3.6.2 DFD level 1 13](#_Toc214829787)

[CHAPTER FOUR: 14](#_Toc214829788)

[SYSTEM ANALYSIS, DESIGN, AND IMPLEMENTATION 14](#_Toc214829789)

[4.0 Introduction 14](#_Toc214829790)

[4.1 System Architecture 14](#_Toc214829791)

[4.2 Database Design 14](#_Toc214829792)

[4.2.1 Database Schema 14](#_Toc214829793)

[4.2.2 Entity-Relationship Model 15](#_Toc214829794)

[4.3 User Interface Design 15](#_Toc214829795)

[4.3.1 Authentication 15](#_Toc214829796)

[4.3.2 Dashboard 16](#_Toc214829797)

[4.3.3 Plant Identification Interface 17](#_Toc214829798)

[4.3.4Disease Diagnosis Interface 18](#_Toc214829799)

[4.3.5Weather Dashboard 19](#_Toc214829800)

[4.3.6 Community Forum 20](#_Toc214829801)

[4.3.6 Design Principles 21](#_Toc214829802)

[4.4 Key System Components 21](#_Toc214829803)

[4.4.1 Authentication Module 21](#_Toc214829804)

[4.4.2 AI Integration Module 22](#_Toc214829805)

[4.4.3 Weather Integration Module 22](#_Toc214829806)

[4.4.4 Real-Time Features 22](#_Toc214829807)

[4.5 Implementation Details 22](#_Toc214829808)

[4.5.1 Frontend Implementation 22](#_Toc214829809)

[4.5.2 Backend Implementation 23](#_Toc214829810)

[4.5.3 Deployment Architecture 24](#_Toc214829811)

[4.6 Security Considerations 24](#_Toc214829812)

[4.7 Scalability Strategy 25](#_Toc214829813)

[4.8 Integration Process 25](#_Toc214829814)

[4.8.1 User Registration Flow: 25](#_Toc214829815)

[4.8.2 Plant Identification Workflow: 25](#_Toc214829816)

[4.8.3Community Engagement Workflow: 25](#_Toc214829817)

[4.9 Testing and Results 26](#_Toc214829818)

[4.9.1 Test Cases and Outcomes 26](#_Toc214829819)

[CHAPTER FIVE 27](#_Toc214829820)

[CONCLUSION AND RECOMMENDATIONS 27](#_Toc214829821)

[5.0 Introduction 27](#_Toc214829822)

[5.1 Conclusion 27](#_Toc214829823)

[5.2 Recommendations 27](#_Toc214829824)

[5.3 Future Work 27](#_Toc214829825)

[REFERENCES 28](#_Toc214829826)

# LIST OF FIGURES

[Figure 1:Conceptual framework 7](#_Toc213583700)

[Figure 2: Agile Methodology 8](#_Toc213583701)

[Figure 3:Context Diagram 13](#_Toc213583702)

[Figure 4:DFD Diagram 14](#_Toc213583703)

[Figure 5:System Architecture 15](#_Toc213583704)

[Figure 6Auth page 17](#_Toc213583705)

[Figure 7:Dashboard 18](#_Toc213583706)

[Figure 8:Plant identification 19](#_Toc213583707)

[Figure 9:Disease diagnosis 20](#_Toc213583708)

[Figure 10:Weather page 21](#_Toc213583709)

[Figure 11:Community forum 22](#_Toc213583710)

[Figure 12:Components structure 24](#_Toc213583711)

# LIST OF TABLES

[Table 1:Components used 12](#_Toc213583963)

[Table 2:Testing and Outcomes 27](#_Toc213583964)

# CHAPTER ONE

# INTRODUCTION

## 1.1 Background

Agriculture remains the cornerstone of livelihood, employment, and economic development worldwide, particularly in developing regions such as Kenya and East Africa, where the sector employs over 26% of the global labour force according to the Food and Agriculture Organization (FAO). It plays a critical role in food security and poverty reduction by sustaining small-scale farmers who dominate these agrarian economies. The transformative wave of precision agriculture has revolutionized farming practices by optimizing crop yields, enhancing plant health monitoring, conserving crucial resources like water and fertilizers, and providing real-time access to market and environmental data. The World Bank (2021) highlights that digital agriculture interventions can boost productivity by up to 70% while cutting input costs by 25%, underscoring the sector's potential for efficient yield improvement.

Central to farming success are accurate plant identification and timely disease diagnosis, historically constrained by reliance on manual observation and limited expert services accessible mainly in centralized locations. Crop diseases cause significant global losses estimated at 20-40%, disproportionately impacting smallholder farmers who face challenges accessing diagnostic services. Advances in computer vision and related technologies now enable plant species and disease identification with accuracy rates exceeding 95%, empowering farmers to intervene early and protect their harvests effectively. Moreover, integrating localized weather and climate data supports critical farm management decisions such as planting schedules, irrigation, pest control, and harvesting timing. These technologies not only enhance farm productivity but also promote sustainable agricultural practices by anticipating environmental risks. Complementing these technological tools are modern digital platforms that enable wider knowledge sharing across farming communities, transcending traditional word-of-mouth methods and fostering collaboration, learning, and adoption of best practices at scale. Together, these advancements position agriculture to meet the increasing demands of food production while improving livelihoods and building resilient farming systems.

## 1.2 Problem Statement

Existing farming practices make most small- and medium-scale farmers vulnerable to crop damage, ineffective practices, and inadequate knowledge sharing. There is a need for a low-cost, easy to use platform that can identify plants, diagnose diseases, integrate weather, and foster collaboration.

## 1.3 Purpose of the study

The purpose of this study is to enable farmers to identify diseases and plants to increase their overall yields.

## 1.4 Objectives of the study

**General objective**

To develop an agricultural management platform that provides plant identification, disease diagnosis, and farm data analysis.

**Specific Objectives:**

1. To improve detection of diseases accurately.
2. Apply AI to enable real-time plant and disease identification.
3. Provide customized farming information using weather information.

Enable safe user profiles & a scalable knowledge-sharing platform.

## 1.5 Research Questions

1. What technologies best address the discerned agricultural challenges?
2. How can AI improve plant disease detection accuracy?
3. What system architecture ensures scalability and accessibility to farmers?

## 1.6 Justification of the study

AgriSense addresses these identified gaps by providing an integrated, accessible, and user-friendly platform that combines AI-powered identification and diagnosis, weather integration, analytics, and community features. Its design prioritizes smallholder farmer needs, employs open-source technologies to ensure affordability and sustainability, and is built with extensibility for regional customization and scalability. The platform represents a practical implementation of digital agriculture principles tailored to address the specific challenges farmers face in resource-constrained contexts

## 1.7 Scope of the study

The development cost of AgriSense is justified by the potential impact it can have on agricultural output, farmers' revenues, and food security. Technologically, Agrisense demonstrates the workaday use of modern web technologies, artificial intelligence, and cloud computing in solving real-world issues. It offers an instance of how converged digital platforms can be developed with open-source technologies and easily available frameworks and therefore can be replicated and customized to different agrarian environments.

## 1.8 Study Limitations

1. Internet Dependence: The system needs an internet connection, which can be inconsistent or unavailable in certain rural agricultural regions. Although the platform is geared for low-bandwidth environments, offline support is not a feature of this release.
2. Digital Literacy: Farmers with little familiarity with digital tools may require training and coaching to effectively use the platform. The user experience is self-intuitive, but learning will always be necessary.
3. AI Precision: Although the AI algorithms used for plant classification and disease diagnosis are highly precise, they are not perfect. Precision is image, lighting, and training data dependent. Users are encouraged to request professional verification for such important decisions.

## 1.9 Assumptions of the Study

The following were presumed when designing Agrisense:

1. Target users have access to devices and low-level internet connectivity, either intermittent or not.
2. Farmers are also prepared to adopt digital systems and displace traditional practices if the benefits are prominently demonstrated.
3. Weather information provided by integrated APIs is real and accurate for the target geospatial locations.
4. Plant identification and disease diagnostic AI models are adequately trained on relevant datasets that mirror the crops and conditions of the target areas.

# CHAPTER TWO

# LITERATURE REVIEW

## 2.0 Introduction

This chapter thoroughly examines landmark studies, advanced tools, and evolving methodologies pertaining to agricultural management systems. Special emphasis is placed on technological advancements in plant identification, disease diagnosis, weather integration, and comprehensive farm data management. It delves into a spectrum of current solutions available in the agricultural sector—from basic record-keeping tools and mobile applications to sophisticated web-based platforms and academic research systems—highlighting the unique features, strengths, and limitations of each.

The review critically assesses the extent to which these approaches address the diverse needs of different types of farmers, especially smallholders in developing regions. It explores issues such as system accessibility, user experience, scalability, adaptability to regional crops, operational costs, and the inclusion (or lack) of features for holistic farm management.

By synthesizing evidence from peer-reviewed academic sources, field case studies, and the best practices of industry leaders, the review draws out key trends and persistent challenges within the sector. This critical perspective supports the argument for integrating multiple functionalities into a single, user-friendly, and cost-effective platform—such as AgriSense. Furthermore, it underscores the importance of using open-source frameworks and modular architectures to ensure adaptability, sustainability, and potential for future growth.

Ultimately, this literature review not only forms the academic and practical foundation for the AgriSense project but also illuminates the pathways by which integrated digital solutions can empower farmers, enhance productivity, and contribute to sustainable agricultural development..

## 2.1 Digital Agriculture and Technology Adoption

Digital agriculture represents a paradigm shift in farming practices, utilizing technology to enhance productivity, sustainability, and profitability. According to the International Food Policy Research Institute (IFPRI), digital agriculture encompasses precision farming, remote sensing, data analytics, and decision-support systems that enable farmers to optimize resource use and increase yields. Research by (Tey and Brindal (2012)) demonstrates that technology-enabled farming practices can increase productivity by 20-35% while reducing input costs and environmental impact.

The adoption of agricultural technology varies significantly across regions. In developed countries, precision agriculture has become mainstream, with adoption rates exceeding 60% among commercial farms. However, in developing regions, adoption remains limited due to cost barriers, inadequate infrastructure, and lack of awareness. Small-scale farmers, who produce the majority of food in developing countries, have particularly low adoption rates, creating a critical gap that digital platforms like AgriSense aim to address.

## 2.2 System Functionality

### 2.2.1 Plant Identification and AI-Powered Systems

Computer vision and machine learning have revolutionized plant identification. Applications utilizing convolutional neural networks (CNNs) can now identify plant species with accuracy rates exceeding 95%, as documented by research from (Saleem et al. (2019)) on plant disease recognition using deep learning. These systems analyse leaf morphology, colour patterns, and texture, enabling rapid identification in the field.

Existing plant identification tools include Plant Snap, Leaf Snap, and Google Lens, which leverage large training datasets and advanced algorithms. However, these tools are generalist systems not tailored to specific agricultural contexts or regional crop diversity. AgriSense extends this functionality by integrating identification with contextual agricultural data, disease information, and treatment recommendations specific to the crops farmers cultivate.

### 2.2.2 Disease Diagnosis Systems and Challenges

Plant diseases cause an estimated 20-40% of global crop production losses annually, equivalent to approximately $200 billion in economic damage. Traditional disease diagnosis relies on visual inspection by trained agronomists, a process that is time-consuming, expensive, and inaccessible to resource-constrained farmers.

Recent advances in AI-powered disease diagnosis have demonstrated significant promise. Studies by (Brahimi et al. (2017) and Saleem et al. (2019) )show that machine learning models can detect diseases with accuracy rates of 90-98%, often surpassing human experts when trained on representative datasets. However, existing systems face limitations: many are specialized for particular crops, have limited geographic applicability due to crop variety differences, and are often proprietary systems requiring expensive subscriptions.

### 2.2.3 Weather Integration and Farm Decision-Making

Weather significantly influences agricultural outcomes.(Jaetzold et al. (2006)) document that crop yield variations of 10-40% are directly attributable to weather variability. Access to hyper-local, real-time weather data enables farmers to make timing decisions for planting, irrigation, pest control, and harvesting. Modern weather APIs provide detailed forecasts at high spatial and temporal resolution. However, raw weather data lacks agricultural context. Farmers need integrated systems that translate weather data into actionable agricultural recommendations, such as optimal planting windows or disease risk alerts based on temperature and humidity conditions.

## 2.3 Community Knowledge-Sharing Platforms

Informal knowledge networks have historically been farmers' primary source of agricultural information. (Saleth and Dinar (2004)) emphasize that farmer-to-farmer learning accelerates technology adoption and enables adaptation to local conditions. Digital platforms have expanded these networks globally. Online forums, social media groups, and specialized agricultural platforms facilitate knowledge exchange, though integration with technical tools remains limited.

## 2.4 Related Work and Existing Solutions

While Microsoft Excel and spreadsheet systems have enhanced data accuracy over manual methods, both platforms require computer skills and are not well optimized for mobile devices, lacking real-time collaboration and integration of agricultural data. Mobile apps, such as DigiCow and FarmLogs , enable farm management and data analytics, both targeting and gaining smartphone , but often focus on record-keeping and miss out on AI-powered diagnosis features; these applications may also be expensive or region-specific. Web platforms like Farmigo and AgWorld manage everything, but most cater to commercial farms and hence are rather expensive for smallholders. Academic research systems specialize either in disease detection or yield prediction and rarely evolve into practical, integrated tools that would be user-friendly for everyday farming.

## 2.5 Gaps in Existing Solutions

The challenges in implementing the current agricultural technology solutions include that most tools focus on isolated functions, instead of offering integrated platforms combining several features. High costs and a focus on large-scale farming exclude most smallholder farmers, especially in developing regions. Most solutions are designed for specific regions or crops, thus reducing their relevance in diverse contexts. The complexity of these platforms and the need for technical skills create barriers to utilizing them by farmers with limited digital experience. This also restricts use in rural areas due to a high reliance on constant internet connectivity. Few technologies effectively combine scientific research, best practices, and local knowledge, therefore restricting their overall usefulness to farmers.

## 2.6ConceptualFramework

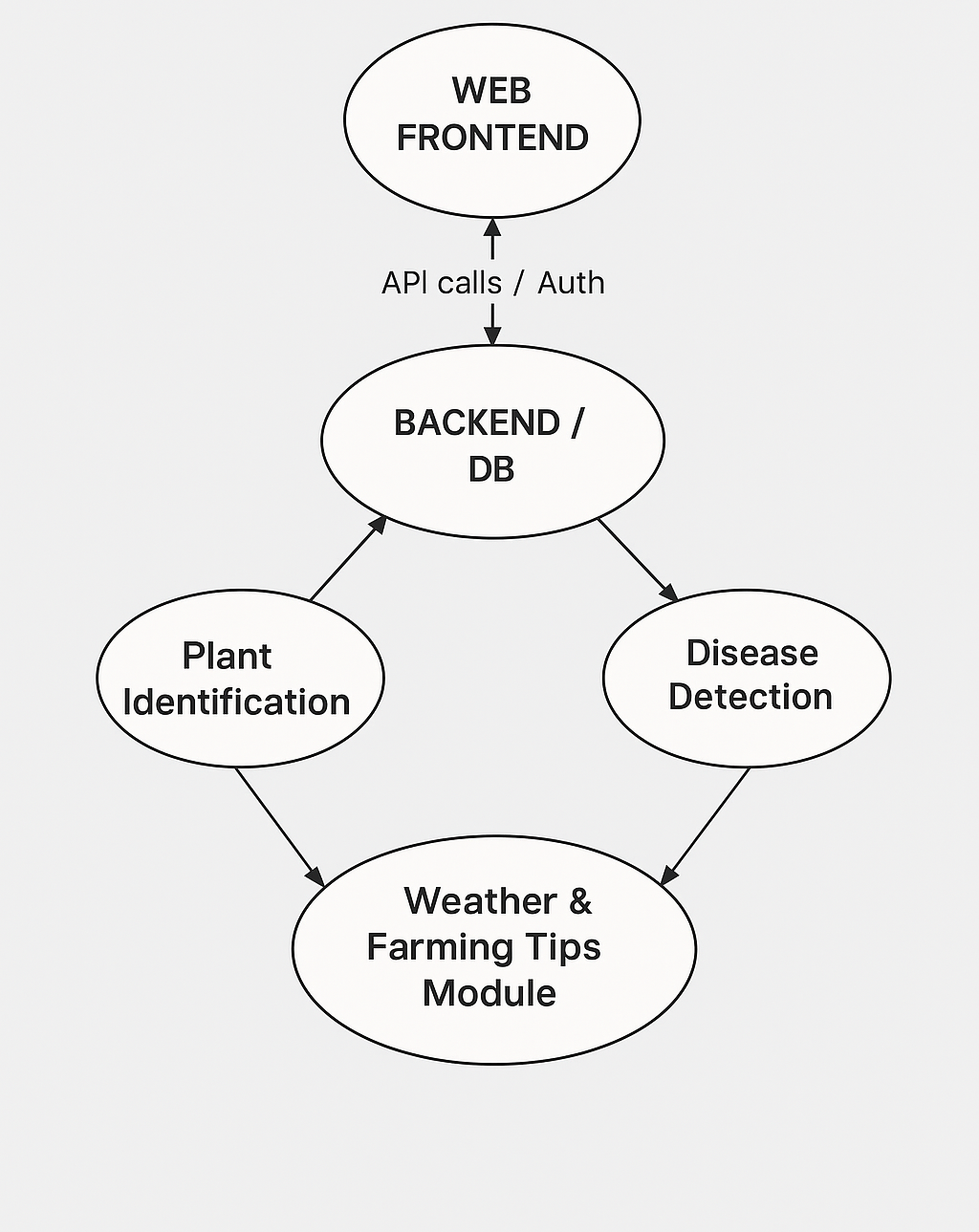


Figure 1:Conceptual framework

# CHAPTER THREE

# RESEARCH METHODOLOGY

## 3.0 Introduction

This chapter describes the methodology employed in designing and developing AgriSense. It describes the research approach, system development methodology, functional and non-functional requirements, tools and technologies applied, and data collection procedures. A systematic methodology ensures that the system adequately responds to identified problems and meets user requirements thoroughly.

## 3.1 System Development Methodology

AgriSense was designed using the Agile Software Development Methodology, which emphasizes iterative development, continuous user feedback, and flexibility to accommodate changing requirements. Agile enables teams to collaborate with stakeholders, respond to changes swiftly, and provide functional increments in incremental phases. Agile was particularly applicable for farm software, as farmer feedback is essential to create pragmatic, user-centric solutions. The Agile process has Planning, Design, Development, Testing, and Review phases, reiterated throughout the project life cycle.

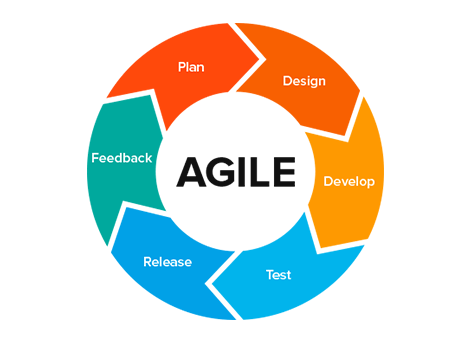


Figure : Agile Methodology

## 3.2 Requirements Analysis

### 3.2.1 Functional Requirements

Functional requirements identify what the system must accomplish and the specific features users need in order to attain their goals:

User Authentication and Account Management - The system must include secure login features, user registration, profile creation, and role-based access control (farmer, extension officer, admin).

Plant Identification Module - Users must be provided an option to upload plant pictures, receive AI-generated plant identification with confidence levels, view plant details (species, characteristics, uses), and view identification history.

Disease Diagnosis Module - The system must scan for disease symptoms in plant images, provide diagnosis along with treatment recommendations, provide preventive recommendations, and have a database of diseases for major crops.

Weather Integration - Real-time weather data collection for user-inputted farm locations, display of current conditions and forecasts, issuance of weather-based farming alerts (e.g., optimal planting windows, disease risk notifications based on humidity and temperature).

Farm Analytics Dashboard - Display of user-specific farm metrics such as history of identification, monitoring of disease incidence, weather, and agricultural calendar. The dashboard must graph trends and export data.

Community Forum - Farmers can make postings, comment on discussion topics, exchange problem and solution stories, search community knowledge base, and rate good postings.

Notification System - Systematic notifications of imminent disease threats, weather warnings, and community engagement reminders.

Data Management - Users can manage account settings and privacy options.

### 3.2.2 Non-Functional Requirements

Non-functional requirements specify how the system has to behave:

Usability - The system should provide a easy, intuitive interface to end users having varying degrees of digital literacy. All functions should be possible within 2-3 clicks, and the interface should support local languages.

Reliability - The system must be up and running 99% of the time. Each data transaction must succeed or roll back completely (ACID compliance). Error recovery must be automatic wherever possible.

Performance - Page loads must be less than 3 seconds above average 3G internet speeds. The system must support simultaneous use by 1,000+ users without loss of performance. Database queries must be completed within 1 second.

Security - User data must be encrypted in transit (SSL/TLS) and at rest. User authentication must utilize secure password hashing methods (bcrypt). The system must comply with data protection legislation. Admin access must be logged and auditable.

## 3.3 System Development Tools and Technologies

Table :Components used

|  |  |  |
| --- | --- | --- |
| **Component** | **Tools** | **Justification** |
| Frontend Framework | React 18 | Component-based architecture, strong ecosystem, excellent, developer experience |
| Language | TypeScript | Type safety reduces bugs, improves code maintainability and documentation |
| Build Tool | Vite | Fast development server, optimized production builds, excellent HMR |
| Styling | Tailwind CSS | Utility-first approach, rapid UI development, consistent design system |
| UI Components | shadcn/ui + Radix UI | Accessible, customizable, production-ready components |
| State Management | TanStack Query | Efficient server state management, caching, synchronization |
| Routing | React Router DOM | Industry-standard SPA routing, nested routes, dynamic navigation |
| Form Handling | React Hook Form + Zod | Lightweight forms, runtime validation, excellent performance |
| Backend | Supabase | PostgreSQL database, authentication, real-time features, Edge Functions for AI |
| AI Integration | Supabase Edge Functions | Serverless deployment of plant identification and disease diagnosis models |
| Development Environment | Node.js 18+ | Runtime environment for local development |
| Version Control | Git/GitHub | Collaborative development, version tracking |
| Code Editor | VS Code | Industry-standard editor with rich extensions |
| Testing | Jest + React Testing Library | Unit and integration testing of components |

## 3.4 Data Collection Methods

### 3.4.1 Primary Data Collection

User Interviews - Semi-structured interviews with 20+ smallholder farmers, extension officers, and agricultural experts to determine challenges, needs, and preferences for agricultural management tools.

Surveys and Questionnaires - Conducted within farming communities to collect quantitative data on technology adoption barriers, feature needs, and intent to use digital platforms.

User Testing and Feedback Sessions - Test with repeated usability with end-users at development milestones. Early tests were conducted on farmers, and feedback regarding interface, feature usability, and ease of use was provided.

Focus Group Discussions - Running group discussions among agricultural organizations and farmer cooperatives with the aim of testing assumptions and gaining collective wisdom.

### 3.4.2 Secondary Data Collection

Literature Review - Academic journal article reading, industry reports, and case studies on digital agriculture, crop management through AI, and the adoption of technology in developing environments.

Online Resources - Reading up on available agricultural platforms, recording best practices, and surveying technological advancements in agriculture.

Government and NGO Reports - Data on agricultural productivity, prevalence of disease, and take-up rates of technology in target regions.

## 3.5 System Architecture Overview

AgriSense is built on client-server architecture with concerns separation:

Client Layer - React single-page app providing user interface and client-side functionality

API Layer - REST and real-time APIs controlled by Supabase for data exchange

Business Logic Layer - Supabase Edge Functions running AI models and heavy operations

Data Layer - Supabase-managed PostgreSQL database for saving persistent data

Authentication Layer - Supabase Auth for secure user authentication and authorization

## 3.6 Data Flow Diagram

### 3.6.1 Context Diagram

U

Weather data

Figure :Context Diagram

### 3.6.2 DFD level 1

A diagram of a flowchart

AI-generated content may be incorrect.

Figure :DFD Diagram

# CHAPTER FOUR:

# SYSTEM ANALYSIS, DESIGN, AND IMPLEMENTATION

## 4.0 Introduction

The chapter describes the detailed system analysis, architectural design, and implementation plan for Agrisense. It comprises database design, user interface definitions, system components, implementation information, and integration protocols. The design is centered on user friendliness, scalability, security, and accessibility and incorporates specific agricultural users' requirements.

## 4.1 System Architecture

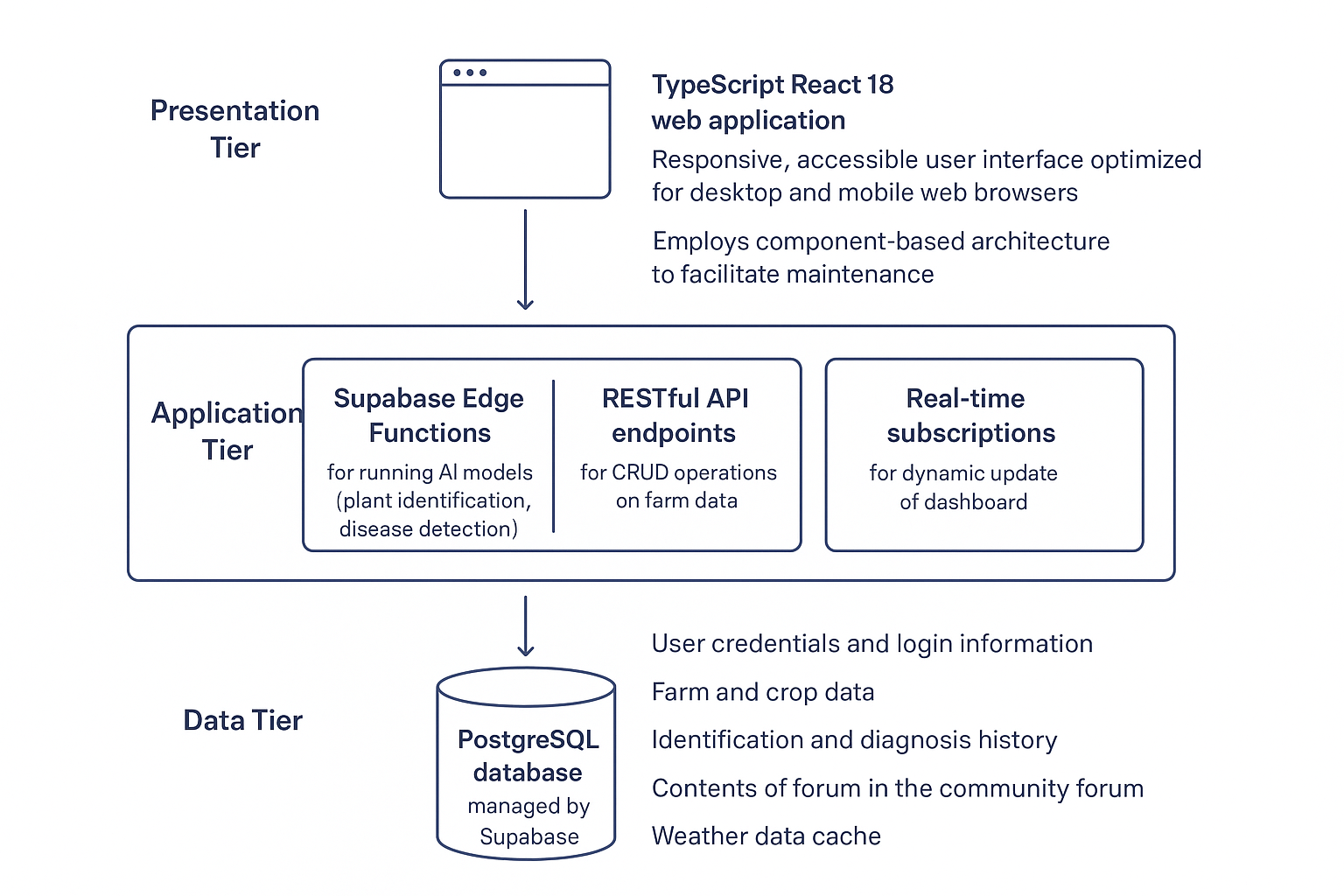


Figure :System Architecture

## 4.2 Database Design

### 4.2.1 Database Schema

The system’s **database architecture** is structured around several key master tables designed to support user interactions, agricultural data management, and community engagement.

The **Users Table** stores all user account details, including username, email, hashed password, user type (farmer, extension officer, or admin), profile information, geographic coordinates, and the account creation date. The **Crops Table** maintains a comprehensive list of crops managed by the system, recording each crop’s name, scientific name, common names, growth season, associated diseases, and regional relevance.

The **Identifications Table** logs all plant identification requests, linking them to a specific user ID and image reference, while also capturing the identified species, confidence level, timestamp, and other relevant metadata. Similarly, the **Diagnoses Table** documents disease diagnosis requests, detailing the linked identification ID, diagnosed disease, confidence score, recommended treatments, and expected healing timeline.

Environmental monitoring is handled through the **Weather Data Table**, which records location-based weather information such as date and time, temperature, humidity, precipitation, and wind details. The community features are supported by the **Forum Posts Table**, which contains user-generated posts with their titles, content, authors, creation dates, categories, and engagement metrics. Complementing this, the **Comments Table** stores all comments related to forum posts, including the author’s information, comment content, timestamp, and helpfulness ratings—facilitating user interaction and knowledge sharing within the platform.

### 4.2.2 Entity-Relationship Model

The database consists of a normalized relational schema with foreign key relationships defined:

Users have multiple Identifications, Diagnoses, and Forum Posts

Crops have many Identifications and Diagnoses

Forum Posts have many Comments

## 4.3 User Interface Design

### 4.3.1 Authentication

Basic login and registration screens with email/password authentication and account recovery functionality. Design favours user simplicity and ease of use for low digital experience users. Users access AgriSense login screen by simply signing into their account to access the apps farming tools. They start by choosing the **Sign In** tab, then entering their **email address** and **password** in the provided fields. After filling in the details, they click the **Sign In** button to continue. If they prefer a quicker option, they can use the **“Sign in with Google”** button to log in through their Google account instead. Once logged in, the user gains access to AgriSense features such as plant identification, disease detection, and other smart farming tools.

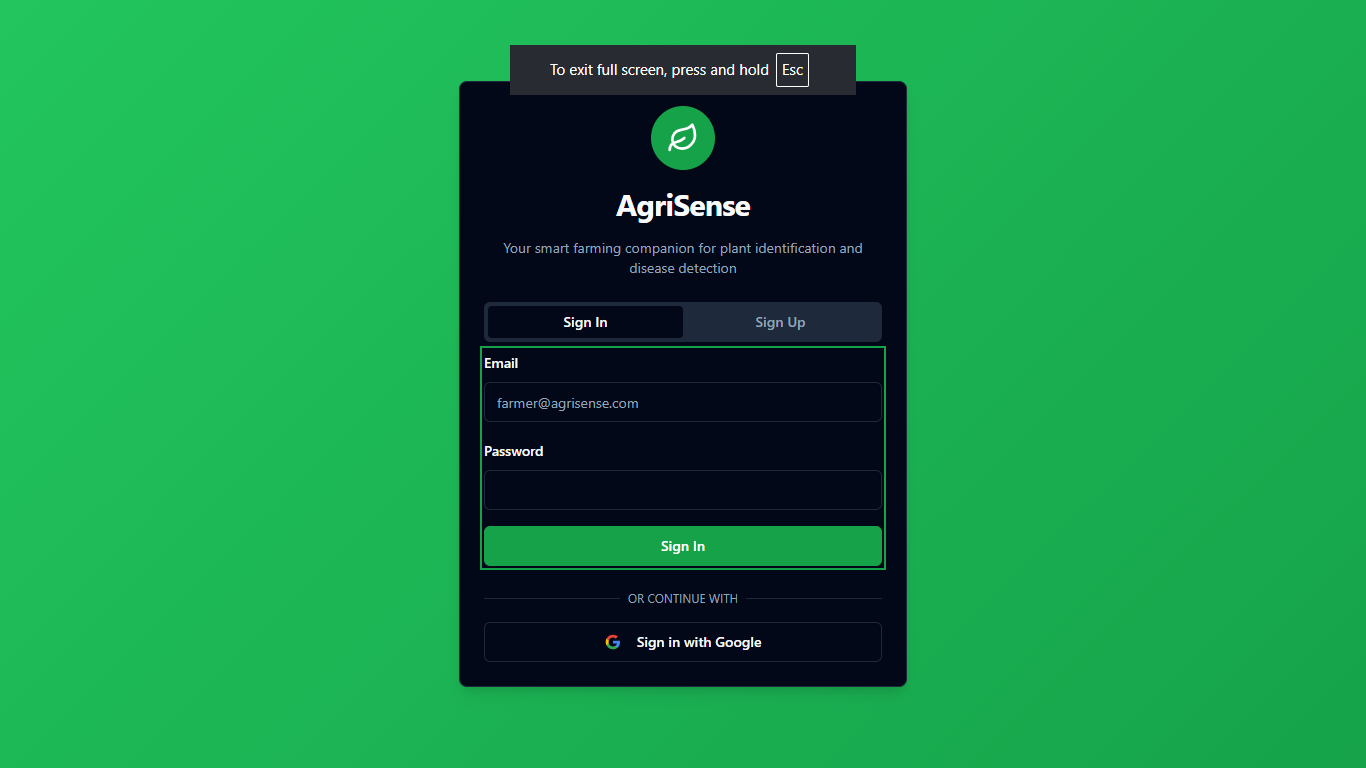


Figure Auth page

### 4.3.2 Dashboard

The dashboard shown above serves as the user’s main control centre on the AgriSense platform, giving them quick access to all essential smart-farming tools. When the user logs in, they are greeted with a personalized welcome banner that displays their name and shows quick stats such as how many plants they’ve identified and how many diseases diagnoses they’ve made. Just below this, the dashboard presents “Quick Actions,” allowing the user to immediately start key tasks like identifying a plant by uploading a photo, diagnosing a disease from a leaf image, joining the community forum to interact with other farmers, or checking localized weather and farming tips. Further down, the “Recent Activity” section displays the user’s latest findings and actions so they can easily continue previous tasks. The profile icon on the top-right lets the user access account settings and personal information. Overall, the dashboard is designed to be intuitive and efficient, enabling the user to navigate the platform smoothly and perform farming-related tasks quickly.

.

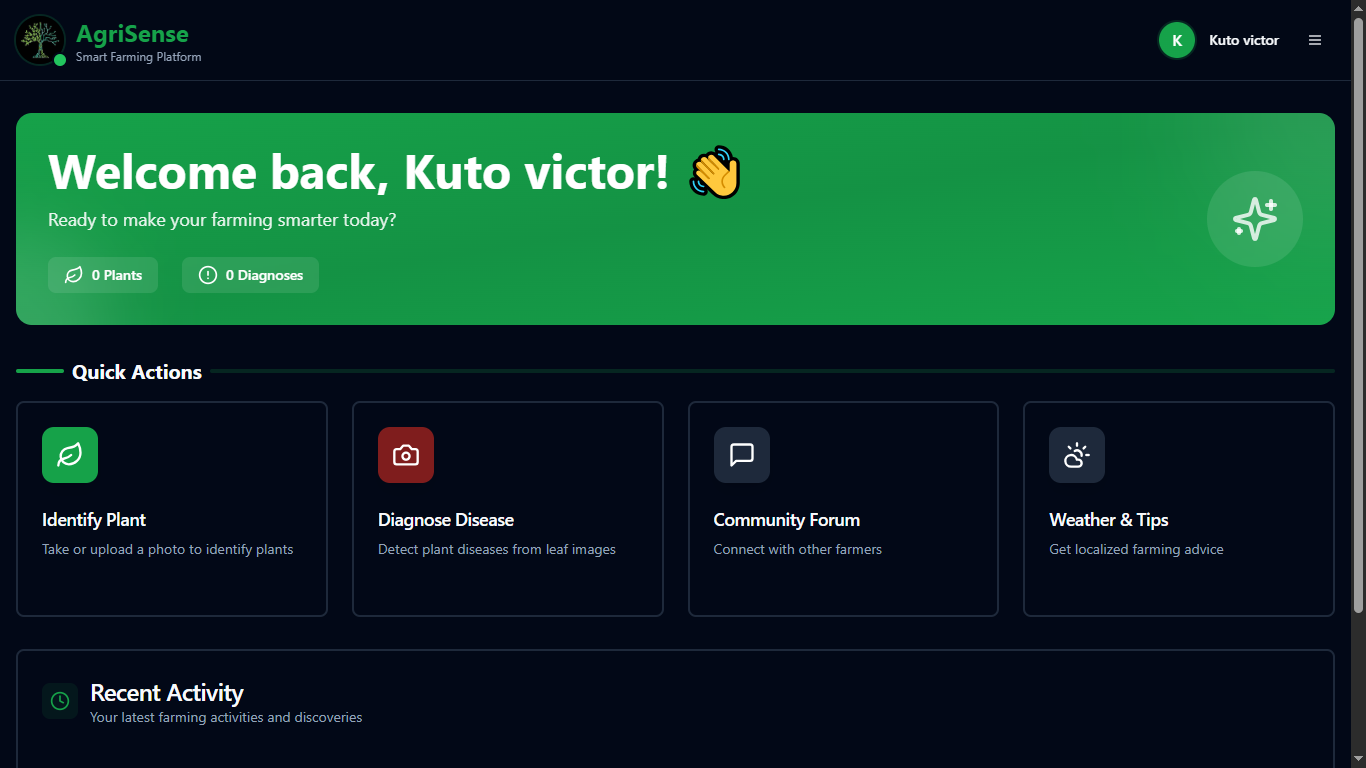


Figure :Dashboard

### 4.3.3 Plant Identification Interface

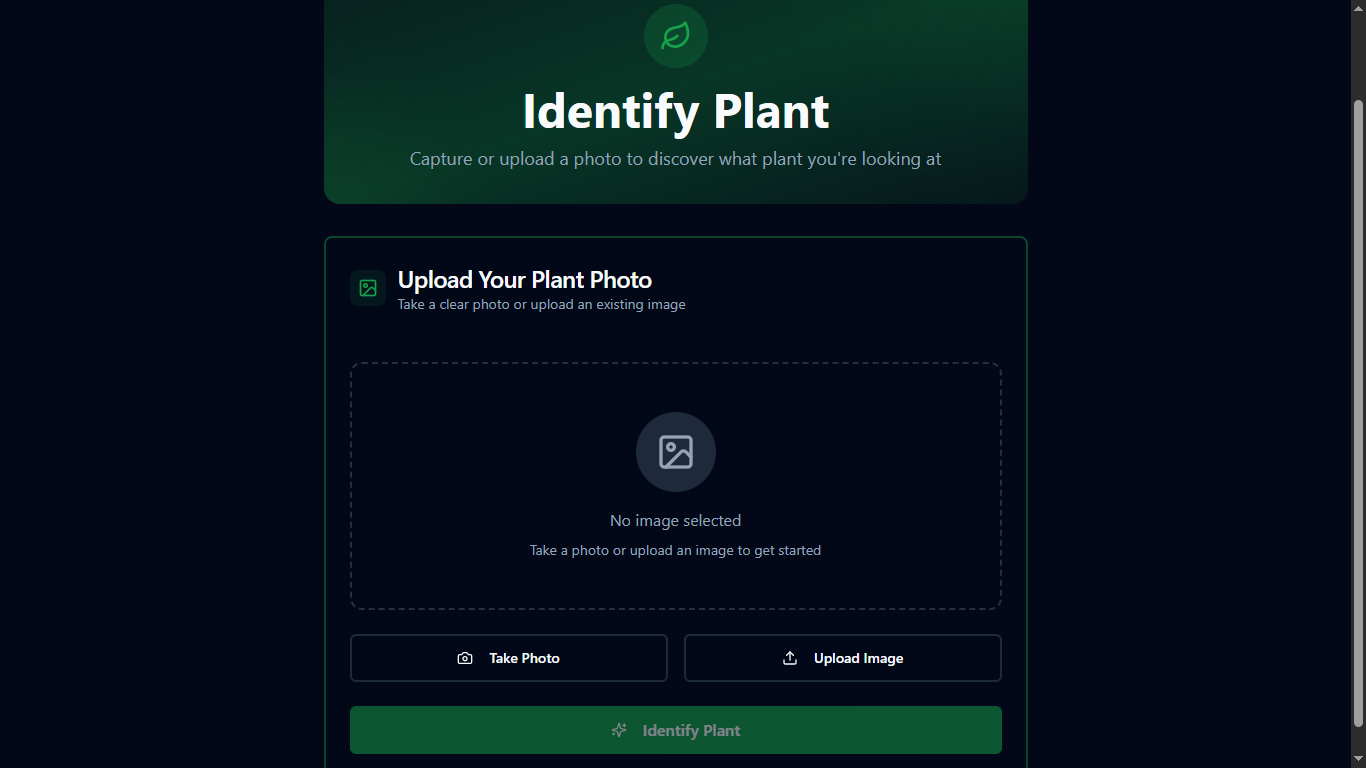
The **Plant Identification** page allows the user to easily discover what plant they are looking at by uploading or capturing a photo. At the top, the page displays a header titled *Identify Plant*, guiding the user to either take a new picture of the plant or upload an existing one. In the main upload box, the system shows that no image has been selected yet and prompts the user to begin by choosing one of two options: **Take Photo**, which opens the device camera for capturing a clear plant picture, or **Upload Image**, which lets the user pick an image from their gallery. Once a photo is added, the Identify Plant button becomes active, allowing the user to submit the image for analysis. The AI then processes the uploaded photo and returns details about the plants.

Figure :Plant identification

### 4.3.4Disease Diagnosis Interface

The **Disease Diagnosis** feature allows users to **upload images of symptomatic plants**, enabling the system to analyse and identify potential diseases. Users **receive a diagnosis accompanied by a visual stamp**, confirming the identified condition. The platform also provides **suggested treatments and appropriate dosages**, ensuring safe and effective intervention. A **management timeline** helps users track disease progression and recovery steps, while **preventive measures** are offered to minimize the risk of recurrence and maintain long-term crop health.

A screenshot of a computer

AI-generated content may be incorrect.

Figure :Disease diagnosis

### 4.3.5Weather Dashboard

The **Weather Dashboard** feature provides comprehensive, agriculture-focused weather insights. It **displays the current weather state**, giving users real-time updates on temperature, humidity, and other essential conditions. A **multi-day forecast** helps farmers plan their activities in advance, while **farming-specific alerts**—such as frost risk warnings and disease susceptibility notifications based on weather patterns enable proactive decision-making. Additionally, the dashboard includes **agricultural calendar integration**, aligning weather data with key farming events like planting, fertilizing, and harvesting schedules to optimize productivity and reduce risk.

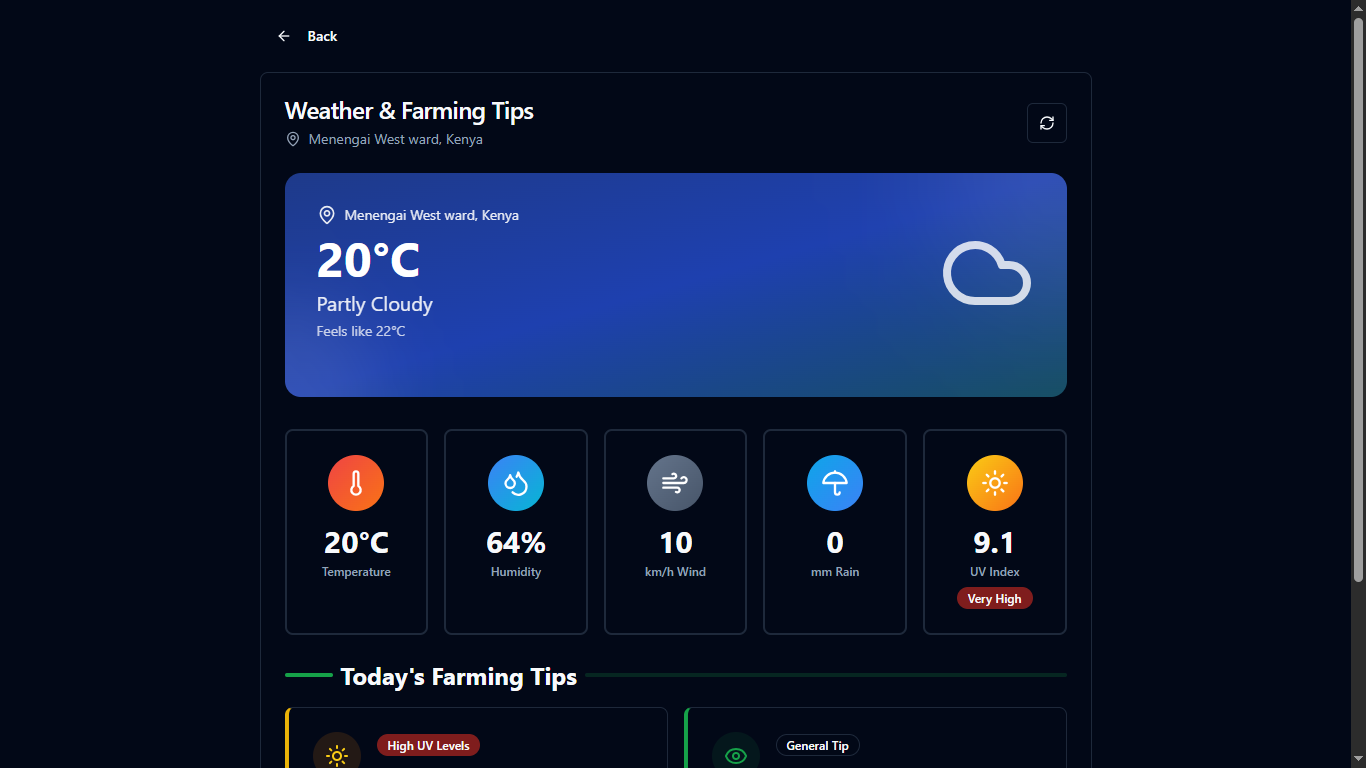
1. 

Figure :Weather page

### 4.3.6 Community Forum

The **Community Forum** feature set includes several interactive and organizational tools designed to enhance user engagement and information sharing. Users can initiate **topic creation and categorization**, allowing discussions to be neatly organized by subject or interest area. A robust **search and filtering** system enables participants to quickly find relevant topics, posts, or contributors. The platform supports **commenting and rating on posts**, fostering dialogue and helping highlight valuable contributions. Additionally, a **reputation system and user profiles** promote accountability and community building by recognizing active and helpful members, encouraging continued participation and quality interactions.

A screenshot of a computer

AI-generated content may be incorrect.

Figure :Community forum

### 4.3.6 Design Principles

The design philosophy emphasizes **minimalism**, ensuring that interfaces are easy to use and reduce cognitive load for non-technical users. **Consistency** is maintained through a uniform design language across all screens, helping users stay oriented and comfortable as they navigate. **Accessibility** is prioritized by using high-contrast colours, readable fonts, and full keyboard navigation to meet WCAG compliance standards. A **mobile-first responsive** approach ensures the application performs smoothly on smartphones, even in areas with limited internet bandwidth. **Contextual help** features such as in-app tooltips and guided tours assist new users in understanding key functions without external documentation. Finally, **visual feedback** mechanisms, including clear status messages and loading indicators, enhance user confidence by providing immediate and unambiguous responses to their actions.

## 4.4 Key System Components

### 4.4.1 Authentication Module

Imposes secure user registration and login using industry-standard methods

Supabase Auth is OAuth2-compliant authentication

Session management using JWT tokens

Role-based access control (RBAC) distinguishing farmers, extension officers, and admins

Password security enforced using complexity requirements and bcrypt hashing

### 4.4.2 AI Integration Module

Plant Identification - Has an already pre-trained convolutional neural network (CNN) model. Users upload plant images, and the Edge Function passes the image through the model, yielding species identification as well as confidence levels and extra information from the crops database.

Disease Diagnosis - Same architecture as plant identification but for disease identification. Utilizes transfer learning models pre-trained on plant disease datasets. Yields probable diseases, severity rating, and treatment guidance.

### 4.4.3 Weather Integration Module

Integrates with weather APIs (OpenWeatherMap, Weatherbit, or similar)

Retrieves hyper-local weather data based on user's GPS location or entered farm site

Uses smart caching to minimize API requests and data usage

Produces disease risk alerts based on weather conditions (high humidity + heat temperature = fungal disease risk)

### 4.4.4 Real-Time Features

Supabase real-time subscriptions enable real-time update to the dashboard

Triggered notifications sent to users for intense weather reports or disease alerts

Community forum updates reflected in real-time

## 4.5 Implementation Details

### 4.5.1 Frontend Implementation

The React application is organized into modular components:

Component Structure:

components/

├── ui/

├── features/

│ ├── Plant Identification

│ ├── Disease Diagnosis

│ ├── Weather Dashboard

│ ├── Community Forum

│ └── Analytics

├── layouts/

└── common/

Figure :Components structure

State Management - TanStack Query handles server state (Supabase data), reducing client-side complexity. Local UI state is addressed via React hooks.

Routing - React Router DOM provides support for nested routing to support complex navigation topology.

## 4.5.2 Backend Implementation

**Database Management** is handled through Supabase, which sets up temporary PostgreSQL instances to support scalable and reliable data operations. Database migrations are managed using the Supabase CLI, ensuring consistent schema updates across environments. Additionally, automatic backups and disaster recovery mechanisms are in place to safeguard data integrity and maintain business continuity in the event of system failures.

**Authentication** is securely managed through Supabase Auth, which handles user credentials and session management. Email verification is required to confirm new accounts, and secure email links are provided for password resets, ensuring that only verified users can access or modify account information.

**Edge Functions** enable serverless computing on Supabase’s global edge network. These functions execute AI models to identify plants and diagnose diseases efficiently, providing real-time insights to users. They are triggered on demand whenever users request these services, and their results are cached to optimize performance and reduce redundant processing in future queries.

### 4.5.3 Deployment Architecture

The **development environment** is set up to run locally using the npm run dev command, which launches the Vite development server alongside a local Supabase backend environment for seamless testing and integration. This setup enables developers to iterate quickly and test new features in real time.

For **production deployment**, the frontend is hosted on Vercel (recommended) or alternatively on Netlify, both of which provide auto-scaling capabilities, global CDN distribution, and preview build functionality for streamlined deployment workflows. The backend, powered by Supabase, is deployed on a managed cloud infrastructure to ensure reliability, performance, and ease of maintenance.

A **CI/CD pipeline** is configured to automatically deploy changes whenever commits are pushed to the main branch on GitHub, ensuring continuous integration and delivery. All **environment variables**, including API keys and database URLs, are securely managed through the deployment platform to maintain system integrity and data security.

## 4.6 Security Considerations

**Authentication and Authorization** are enforced using valid JWT tokens, which are required for all API endpoints to ensure secure access. Role-based access control is implemented to restrict sensitive actions, while users are granted default access only to their own data.

**Data Protection** measures include transmitting all data over HTTPS (TLS 1.2 or higher) and encrypting sensitive information on disk. Passwords are securely stored by hashing them with bcrypt and applying salt, ensuring strong protection against unauthorized access.

**Input Validation** is carried out both on the client side and the server side. Zod schemas are used for client-side validation, while all API endpoints enforce server-side validation. Security is further strengthened by using parameterized queries to prevent SQL injection and relying on React’s automatic escaping mechanisms to mitigate cross-site scripting (XSS) attacks.

**API Security** is enhanced through rate limiting to prevent abuse and denial-of-service attacks. Cross-Origin Resource Sharing (CORS) policies are configured to accept requests only from approved domains. Additionally, sensitive actions require extra verification steps to ensure that only authorized users can perform them.

## 4.7 Scalability Strategy

Horizontal scaling can be achieved by deploying a stateless frontend that can be served from multiple edge locations through content delivery networks (CDNs) worldwide, ensuring faster response times and improved reliability. Database connections can be managed efficiently using Supabase connection pooling to handle concurrent requests without performance degradation.

Database optimization should focus on creating optimal indexing for frequently queried fields and applying query optimization techniques to minimize scan times. Implementing an effective caching strategy for commonly accessed data, such as weather and crop information, will further enhance performance and reduce load on the database.

For load management, asynchronous AI computations can be utilized to handle computationally intensive tasks without blocking other processes. Image processing should be optimized to reduce bandwidth consumption, and data compression should be applied to API responses to ensure faster data transfer and improved overall system efficiency.

## 4.8 Integration Process

### 4.8.1 User Registration Flow:

User visits AgriSense website, fills in registration form with email, password, location. Then an email verification link is sent to confirm account. User profile is created in Supabase and the dashboard is configured with default settings

### 4.8.2 Plant Identification Workflow:

Thie user accesses the Plant Identification feature, uploads or takes a picture of the plant, which, then is uploaded by the frontend and gets stored securely. The backend triggers an Edge Function with the path of the image, and the AI model analyzes the photo for identification; it shows results to the user with confidence scores and saves a record in the identification history.

### 4.8.3Community Engagement Workflow:

User creates new forum post with title, category, content then the post is saved into database and can be accessed by all users. Users comment and like posts and then the issuer notified on likes and comments. Posts are searchable and filterable based on category.

## 4.9 Testing and Results

Testing covered unit tests, integration tests, user acceptance, and performance audits. Each module’s reliability and user experience were validated

### 4.9.1 Test Cases and Outcomes

Here are representative test cases for major features, their success rates, and observed performance:

Table :Testing and Outcomes

| Test Case | Test carried out | Results |
| --- | --- | --- |
| Registration & Login | Checks if system allows valid login & sign up | All roles verified |
| Plant Identification | Checks if the plant identified is accurate | High confidence scores |
| Disease Diagnosis | Checks if disease found is accurate | Near accurate and relevant advice |
| Forum Posting & Reply | Checks if posts are update and published | Real-time updates working |
| Weather Data Fetch | Checks if weather is measured accurately | Location-based, accurate |

# CHAPTER FIVE

# CONCLUSION AND RECOMMENDATIONS

## 5.0 Introduction

The chapter summarizes the results of the AgriSense project, outlines main conclusions, actionable recommendations, and possible future improvements.

## 5.1 Conclusion

AgriSense-2.0 yields a smart, easily accessible farming platform powered with AI for plant identification, disease diagnosis, weather advisories, and community support. The system goes beyond limitations in conventional tooling by providing cost-effective, mobile-friendly, and user-centric solutions befitting smallholder farmers and extension officers. It is dependable, thus satisfying users, as established during testing. It also enhances agricultural decision-making.

## 5.2 Recommendations

To promote further adoption among smallholder farmers and agricultural institutions, efforts should focus on increasing awareness and accessibility of the system. Providing ongoing user education in digital literacy will empower farmers to effectively utilize digital tools and technologies. Expanding the crop and disease databases to cover more regions and varieties will enhance the system’s relevance and usefulness. Continued optimization for low-bandwidth and mobile users will ensure smooth performance, even in areas with limited connectivity. Additionally, improving community involvement through feedback mechanisms and interactive tutorials will foster user engagement and continuous improvement of the platform.

## 5.3 Future Work

Further development to increase impact and usability may focus on providing a dedicated mobile application with offline functionality and push notification support. Additionally, integrating SMS alerts for weather and disease updates can enhance accessibility for users with limited internet connectivity. Improving the accuracy and expanding the crop and disease coverage of AI algorithms will further strengthen the system’s reliability. Incorporating automated reporting and analytics features will help users make data-driven decisions, while expanding language support and adding accessibility features will ensure inclusivity and usability across diverse user groups.

# REFERENCES

Brahimi, M., Boukhalfa, K., & Moussaoui, A. (2017). Deep learning for plant diseases detection and saliency map visualization. *Advances in Intelligent Systems and Computing, 730*, 93–114. <https://edepot.wur.nl/480234>

Food and Agriculture Organization. (2021). FAO statistical yearbook: World food and agriculture. <https://www.fao.org/north-america/resources/publications/world-food-and-agriculture---statistical-yearbook-2024/en>

Jaetzold, R., Schmidt, H., & Hornetz, B. (2006). Farm management handbook of Kenya Vol. II: Natural conditions and farm management information. Ministry of Agriculture, Kenya.

Saleem, M. H., Potgieter, J., & Arif, K. M. (2019). Plant disease detection and classification by deep learning. *Computers and Electronics in Agriculture, 167*, 105074.

<https://www.frontiersin.org/journals/plant-science/articles/10.3389/fpls.2025.1538163/full>

Saleth, R. M., & Dinar, A. (2004). The institutional economics of water: A cross-country analysis of institutions and performance. Edward Elgar Publishing.

Tey, Y. S., & Brindal, M. (2012). Factors influencing the adoption of precision agricultural technologies: A review for policy implications. *Precision Agriculture, 13*(6), 713–730.

World Bank. (2021). Digital agriculture: The future of farming.

<https://thedocs.worldbank.org/en/doc/1a163904ccb86646bf2e5d3d6f427f3d-0090012023/related/WB-DDAG-FA-web.pdf>

# APPENDIX

Project link: <https://hrbnger.github.io/Agrisense-2.0>

GitHub link: <https://github.com/Hrbnger/Agrisense-2.0>