



Instituto Politécnico
de Castelo Branco
Escola Superior
de Tecnologia

DEVELOPMENT OF A SYSTEM FOR IDENTIFICATION AND GEOLOCATION OF DISEASES, PESTS AND ANOMALIES IN AGRICULTURAL CROPS

Mauro De Jesus Manuel Francisco

20210542

Advisors

Professor Doutor **José Carlos Meireles Monteiro Metrôlho**

Professor Doutor **Fernando Reinaldo da Silva Garcia Ribeiro**

Dissertation presented to School of Technology of Polytechnic Institute of Castelo Branco to fulfill the necessary requirements to obtain the Master's degree in Computer Engineering - Specialization Area in Software Development and Interactive Systems, carried out under the scientific guidance of Professor José Carlos M. M. Metrôlho and Professor Fernando Reinaldo Ribeiro, from the Polytechnic Institute of Castelo Branco.

September of 2024

Jury composition

Jury president

Alexandre José Pereira Duro da Fonte, Professor Adjunto da UTC de Informática da Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco.”

Vowels

Carlos Cambra Baseca, Professor Associado, Área de Ciencias de la Computación e Inteligencia Artificial, Departamento de Digitalización, Escuela Politécnica Superior da Universidade de Burgos - Espanha (arguente).”

Ângela Cristina Marques de Oliveira, Professora Adjunta da UTC de Informática da Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco (arguente).”

José Carlos Meireles Monteiro Metrôlho, Professor Coordenador da UTC de Informática da Escola Superior de Tecnologia do Instituto Politécnico de Castelo Branco (Orientador)”

Dedication

I dedicate this work to my parents, who unwaveringly supported me throughout this pivotal phase of my life, and who, with profound affection and empathy, consistently encouraged and motivated me to achieve my goals. To my beloved grandparents, whom I hold dear and cherish, I am confident that regardless of their whereabouts, they will forever be observing and protecting me. They reside within my heart. Perpetual yearning.

Thanks

First and foremost, I would like to express my gratitude to the Creator, who bestows life, for the gift of breath and good health.

It is vital to acknowledge that the development of this project would not have been possible without the invaluable help of certain individuals who directly or indirectly assisted me and played a crucial role in the successful completion of this significant phase in my life.

I would like to sincerely and profoundly thank Professor Doctor José Carlos Meireles Monteiro Metrôlho and Professor Doctor Fernando Reinaldo da Silva Garcia Ribeiro for their invaluable assistance, guidance, availability, and expertise provided during this project. Their exceptional professionalism and understanding have been highly appreciated. Without their presence, none of this would be achievable.

I would like to express my gratitude to my classmates, Luís Fonseca and Leandro Manjate, for their invaluable assistance during this project. They dedicated numerous hours to engaging in fruitful discussions, attentively listening to my concerns, and helping me in identifying potential resolutions.

I would also like to extend my gratitude to all my other acquaintances who provided me with support, encouragement, and inspiration to improve myself and strive towards my aspirations.

Finally, I would like to express my immense gratitude to my parents, as they are the very purpose of my existence and the ultimate culmination of everything.

According to William Marrion Branham, expressing gratitude to God for the small things we possess is the crucial factor in obtaining significant blessings.

Gratitude to everyone!

Abstract

Achieving high production and sustainability in agriculture requires quick detection and control of diseases, pests, and anomalies in agricultural crops. In this dissertation, a system built with machine learning approaches is presented that automatically detects and geolocates these problems in crops. Real-time data collection and analysis are made possible by the combination of a mobile app and a web-based Admin Panel. The main goals of the system are to automatically identify diseases from pictures of plant leaves, to register and geolocate anomalies found, and to give users comprehensive data visualizations. Through the integration of these features, the system provides means that can help in increasing crop yields, promoting sustainable farming methods, and improving agricultural administration.

The development process involved a close examination of the current technologies and approaches, an analysis of the system requirements and a modular and scalable system design and development. The solution uses machine learning models for accurate disease identification together with ASP.NET MVC for web development and Flutter with Dart for Mobile Application development. Farmers and agricultural managers can benefit greatly from the system. Along with advancing agricultural technology, this effort has important ramifications for environmental sustainability and food security.

Keywords

Agricultural Technology, Crop Disease Detection, Machine Learning, Plant Disease Geolocation, Crop Management.

Resumo

Alcançar alta produção e sustentabilidade na agricultura requer a rápida detecção e controlo de doenças, pragas e anomalias nas culturas agrícolas. Nesta dissertação, é apresentado um sistema construído com abordagens de aprendizagem automática, capaz de detetar e geolocalizar automaticamente esses problemas nas culturas agrícolas. A recolha e análise de dados em tempo real são possibilitadas pela combinação de uma aplicação móvel e um Painel de Administração baseado na web. Os principais objetivos do sistema são identificar automaticamente doenças a partir de imagens de folhas das plantas, registar e geolocalizar as anomalias encontradas, e fornecer aos utilizadores visualizações abrangentes dos dados. Através da integração destas funcionalidades, o sistema oferece um meio que pode ajudar a aumentar os rendimentos das culturas, promover métodos agrícolas sustentáveis e melhorar a gestão agrícola.

O processo de desenvolvimento envolveu uma análise dos requisitos do sistema, um design modular e escalável do sistema, e um exame minucioso das tecnologias e abordagens atuais. A solução utiliza modelos de aprendizagem automática para uma identificação precisa das doenças, juntamente com o Flutter com Dart para o desenvolvimento da aplicação móvel e o ASP.NET MVC para o desenvolvimento web. Agricultores e gestores agrícolas podem beneficiar enormemente do sistema. Além de avançar na tecnologia agrícola, este esforço tem importantes implicações para a sustentabilidade ambiental e a segurança alimentar.

Palavras Chaves

Tecnologia Agrícola, Detecção de Doenças em culturas agrícolas, Aprendizagem de Máquina, Geolocalização, Gestão de Culturas

General index

1.	<i>Introduction</i>	<i>1</i>
1.1	Objectives	2
1.2	Planning and Methodology	3
1.3	Document Organization	4
2.	<i>State of the Art.....</i>	<i>6</i>
2.1	Purpose of the Review	7
2.2	Related Work	8
2.3	Methodology.....	11
2.3.1	Search Strategy	12
2.3.2	Screening for Inclusion.....	12
2.3.3	Screening for Exclusion	13
2.3.4	Results Summary.....	13
2.4	Data Extraction and Analysis.....	15
2.5	Discussion	34
2.6	Strengths and Limitations	38
2.7	Related Applications.....	39
2.8	Final Remarks.....	40
3.	<i>Requirements and Modeling.....</i>	<i>42</i>
3.1	Requirements.....	43
3.1.1	Functional Requirements	45
3.1.2	Non-functional Requirements.....	46
3.2	Use Cases Diagrams	47
3.3	Use Cases Description	52
3.3.1	Admin use cases.....	53
3.3.2	Team Leader use cases.....	56
3.3.3	Team Member Use Cases.....	60
3.4	Storyboards.....	65
3.5	Data Model Entity-Relationship	77
3.5.1	Database Relational Model	78
3.5.2	Description of Relational Model Tables	80
3.6	Final Remarks.....	86
4.	<i>Architecture, Tools and Implementations.....</i>	<i>87</i>
4.1	System Architecture	88

4.2	Tools.....	90
4.2.1	Admin Panel.....	90
4.2.2	Mobile App	91
4.3	Implementation	92
4.3.1	Admin Panel.....	92
4.3.2	Mobile App	106
4.4	Tests and Results.....	120
4.5	Final Remarks.....	122
5.	Conclusions.....	124
	References.....	127

Index of figures

Figure 1. The flowchart of the systematic review process determined the papers selected for the present study (adapted from the PRISMA 2020 flow diagram [31]...	14
Figure 2. <i>Top</i>) Most used models. <i>Bottom</i>) Most used models over the years.	38
Figure 3. Admin use cases diagram	49
Figure 4. Team Leader use cases diagram	50
Figure 5. Team Member use cases diagram	51
Figure 6. Storyboard related to the web application registration screen.	66
Figure 7. Storyboard related to the web application login screen.	67
Figure 8. Storyboard related to the web application Admin main screen.....	68
Figure 9. Storyboard related to the web application Team Leader main screen..	69
Figure 10. Storyboard related to the web application Team Member main screen.	70
Figure 11. Storyboard related to the web application Manage Users screen.	71
Figure 12. Storyboard related to the web application Manage Roles screen.....	72
Figure 13. Storyboard related to the web application Create new user screen.....	73
Figure 14. Storyboard related to the web application Create new disease report screen.	74
Figure 15. <i>Storyboard related to the mobile application mixed actions screens.</i>	76
Figure 16. <i>System Data Base Relational Model</i>	79
Figure 17. <i>System Architecture</i>	88
Figure 19. <i>Admin Panel admin home screen</i>	94
Figure 20. <i>Admin Panel admin panel – manage users</i>	95
Figure 21. <i>Admin Panel admin panel – manage roles.....</i>	96
Figure 22. Admin Panel team leader panel – main screen.....	97
Figure 23. Admin Panel team leader panel – teams management	98
Figure 24. Admin Panel team leader panel – teams management – edit team.....	99
Figure 25. Admin Panel team leader panel – teams management – send invite .	100
Figure 26. Admin Panel team leader panel – teams management – send invite successful message.....	100
Figure 27. Member email received from team leader’s invite	101
Figure 28. Admin Panel team leader panel – members management	102
Figure 29. Admin Panel team leader panel – members management – delete action	102
Figure 30. Admin Panel team leader panel – disease reports management	103
Figure 33. App mobile – Login screen	108
Figure 35. App mobile – Disease report menu-option screens	111
Figure 36. App mobile – Disease report creation menu-option screens.....	113
Figure 37. App mobile – Data Visualization navbar screen	114
Figure 38. Teachable Machine Screen – images class creation.....	117
Figure 39. Teachable Machine Screen – model training completed	118

Figure 40. Teachable Machine Screen – model export	119
---	------------

List of tables

Table 1. Scheduling of the various work stages.....	4
Table 2. Studies reviewing the detection, identification, and classification of diseases and pests in agricultural crops.	9
Table 3. Summary of the articles selected for analysis and synthesis.	28
Table 4. Summary of the datasets and algorithms.....	31
Table 5. Users Table description	80
Table 6. Roles Table description.....	80
Table 7. UserRoles Table description	81
Table 8. Teams Table description	81
Table 9. UserTeams Table description	81
Table 10. Diseases Table description.....	81
Table 11. Disease Reports Table description	82
Table 12. Pests Table description	82
Table 13. Pest Reports Table description	83
Table 14. Anomalies Table description.....	83
Table 15. Anomaly Reports Table description.....	84
Table 16. Type Issues Table description	84
Table 17. Recommended suggestions Table description.....	85
Table 18. Suggestion Reports Table description.....	85

List of abbreviations and acronyms

AI - Artificial Intelligence

ANN - Artificial Neural Network

API - Application Programming Interface

CFNN - Cascaded Forward Neural Network

CNN - Convolutional Neural Network

CRUD - Create, Read, Update, Delete

DAML - Deep Adversary Metric Learning

DB - Database

D-CNN - Deep Convolutional Neural Network

DL - Deep Learning

EFFTA - Enhanced Fusion Fractal Texture Analysis

FFNN - Feedforward Neural Network

FPN - Feature Pyramid Network

FSL - Few-Shot Learning

GAN - Generative Adversarial Network

GLCM - Grey-Level Co-occurrence Matrix

GPS - Global Positioning System

GWT - Gray Level Co-occurrence Matrix

HTTP - Hypertext Transfer Protocol

IoT - Internet of Things

JSON - JavaScript Object Notation

k-NN - k-Nearest Neighbors

LeNet - A convolutional neural network structure

LR - Logistic Regression

MIB - Minimal Inhibitory Concentration

ML - Machine Learning

MSCO - Multi-Swarm Coyote Optimization

NASNetMobile - Neural Architecture Search Network Mobile

NBAIR - National Bureau of Agricultural Insect Resources

NoSQL - Non-relational SQL

ORB - Oriented FAST and Rotated BRIEF
PRISMA - Preferred Reporting Items for Systematic Reviews and Meta-Analyses
RESTful - Representational State Transfer compliant web services
RF - Random Forest
RMSprop - Root Mean Square Propagation
RSC - Random Subspace Classifier
SIFT - Scale Invariant Feature Transform
SQL - Structured Query Language
SSL - Secure Sockets Layer
SVM - Support Vector Machine
TCP/IP - Transmission Control Protocol/Internet Protocol
TL - Transfer Learning
ToLCGV - Tomato Leaf Curl Gujarat Virus
ToLCNDV - Tomato Leaf Curl New Delhi Virus
UAV - Unmanned Aerial Vehicle
VGG - Visual Geometry Group
Xception - Extreme Inception
YOLO - You Only Look Once
XML - eXtensible Markup Language

1. Introduction

Plant diseases are considered one of the main factors that influence food production, being responsible for a significant reduction in the physical or economic productivity of crops and, in some cases, can be an impediment to this operation. To maintain control of production losses and maintain the sustainability of crops, it is essential that some measures are carried out properly, such as constant monitoring of the crop, combined with the rapid and accurate diagnosis of the respective diseases, pests or anomalies. These practices are usually recommended by specialists in plant pathology [1].

These challenges are increasing for any agricultural producer, anywhere on the planet. It is important that farmers are aware of such challenges in their operations in good time. Thus, it becomes useful to access technology to help deal with these threats to agriculture, to increase agricultural productivity and operating profits.

Technological progress has enabled the use of techniques and methods capable of optimizing agricultural returns [2], preserve natural resources [3], reduce unnecessary use of fertilizers [4], and identify diseases in crops and animals from remote sensing images [5]. Which circumscribes a great value in terms of monitoring crops on a scale.

This research aims to tackle these issues by developing a prototype system that uses artificial intelligence resources to automatically identify diseases in agricultural activities or expeditions based on leaf photos. The core principle of this prototype is to employ AI and machine learning techniques to accurately detect and classify various agricultural diseases by analyzing photographs of plant leaves. This approach not only enhances the precision of disease identification but also facilitates immediate intervention, which is vital for minimizing crop losses and ensuring sustainable agricultural practices.

Furthermore, the system has an administrative panel that serves as a centralized hub for collecting and supervising all the data gathered from the field. This administrative panel's main objective is to streamline the process for users to submit, store, and analyze data pertaining to discovered diseases, pests, and abnormalities. The admin panel provides comprehensive information on the health state of crops, enabling farmers and agricultural managers to make well-informed decisions. It assists

in visualizing the dispersion of identified issues, understanding their spread over time and space, and developing proactive measures to address them. The goal is to create a robust system that automates the identification process and enhances the overall management of agricultural activities through data-driven insights.

Furthermore, the system utilizes geolocation capabilities to accurately plot the precise positions of recognized problems, facilitating a spatial understanding of the distribution of diseases and pests. This capability is especially valuable for tracking the spread of issues and specifically tailoring actions to the areas where they are required. The capacity to exchange information and monitor past data enables more effective collaboration and well-informed decision-making over time.

The process of collecting data in the field is made easier by using a mobile application. This allows team members to use their smartphones to take pictures of leaves and enter data directly while in the field. This guarantees the collection of data in real-time and the prompt analysis of it, resulting in an efficient procedure and minimizing response delays. By including these sophisticated functionalities, the system offers a comprehensive solution for overseeing the well-being of crops, enhancing agricultural output, and guaranteeing the long-term viability of farming methods.

Essentially, this prototype seeks to incorporate technology into regular agricultural procedures, providing a comprehensive tool for identifying and managing diseases. This tool has the potential to greatly enhance crop yields and guarantee food security.

1.1 Objectives

This work's objective is to develop a system, consisting of a web and mobile application, for the identification and geolocation of diseases, pests and anomalies in agricultural crops. More specifically, the system should allow:

- Automatic detection and identification of diseases from images of plant leaves;
- Registration of diseases, pests and anomalies;
- Geolocation of identified diseases, pests and anomalies;
- Visualization and analysis of the distribution of collected data;
- Customization of data management according to the user profile.

1.2 Planning and Methodology

To achieve the desired objectives, a sequence of six stages is expected, as detailed in the subsequent description and schedule. This work adheres to the Waterfall methodology [6], a widely employed linear and sequential strategy in software development, for its planning and implementation. This technique is appropriate for this project as it guarantees comprehensive documentation and organized advancement through each stage, enabling a methodical development process.

The project consists of six stages, which are outlined below:

- 1) Examine and assess pre-existing systems: This involves studying existing solutions designed with similar goals, particularly those enabling the detection of diseases and the organization of data on abnormalities in agricultural crops, including their geolocation.
- 2) Identify and clearly define the necessary specifications: This step involves detailing the functional and non-functional requirements that the system must meet to achieve its objectives effectively.
- 3) Create a system by formulating and structuring it through modelling and designing: This phase includes creating detailed design documents, use case diagrams, and architectural models that will guide the system's development.
- 4) Solution implementation and testing: In this stage, the actual coding of the system takes place, followed by rigorous testing to ensure all functionalities work as intended and meet the specified requirements.
- 5) Evaluation of the system-wide level of acceptance: This step involves assessing the system's acceptance among potential users and stakeholders, gathering feedback, and making necessary adjustments.
- 6) Compose a written report: A comprehensive report documenting the entire development process, methodologies used, results achieved, and lessons learned will be prepared.

Next, table 1 demonstrates the arrangement of these several work phases. The table delineates the chronological sequence for each phase, guaranteeing a lucid and methodical strategy for carrying out the work. Briefly, the following table visually presents the project timeline, outlining the specific times assigned to each of the six

stages throughout the year. This schedule facilitates the monitoring of progress and guarantees the timely completion of all tasks.

Table 1. Scheduling of the various work stages.

	2022	2023												2024				
	Dec.	Jan.	Feb.	Mar.	Apr.	Mai.	Jun.	Jul.	Aug.	Sept.	Oct.	Nov.	Dec.	Jan.	Feb.	Mar.	Apr.	May
1																		
2																		
3																		
4																		
5																		
6																		

1.3 Document Organization

This report is organized into five chapters, each dedicated to specific, but interconnected, components of the research and development process.

Chapter 1, the “Introduction”, establishes the structure presenting information related to the work being presented. The text explores the significance and extent of the research, outlining the primary objectives, which involve the creation of a system to identify and geolocate diseases, pests and anomalies in agricultural crops. The planning process, which includes important milestones and the overall structure of the document, is also described.

Chapter 2, titled "The State of the Art," provides a thorough review of existing literature and techniques pertaining to the automated identification, diagnosis, and classification of diseases and pests in agricultural crops. The purpose of this chapter is to offer a comprehensive understanding of the most recent breakthroughs and methodologies, which will serve as a basis for creating the proposed system. The chapter comprises comprehensive sections on the review's objectives, relevant research, approach, data retrieval and analysis, discourse, strengths and constraints, and practical implications of the results.

Chapter 3, titled "Requirements and Modeling," provides an overview of both the functional and non-functional requirements that need to be met by the proposed

system. The content comprises use cases illustrations and explanations that represent the interactions among various user roles and the components of the system. This chapter presents a plan for the development of the prototype present in this work, ensuring that all requirements are designed.

Chapter 4, Architecture, Tools, and Implementations, provides an overview of the system's overall structure, which includes its various components and how they interact with each other. This chapter presents the tools used in creating online and mobile components, providing a clear understanding of the purpose and functioning of each tool. The chapter also explores the practical aspects of implementation, providing detailed information on the processes involved in developing the Admin Panel web system and the mobile application.

Chapter 5, Conclusions, provides a summary of the project's main findings and their consequences. It provides a thorough summary of the research and development efforts, including an analysis of the research's impact, potential future work, and suggested areas for improvement. This chapter intends to contemplate the accomplishments of the study and suggest avenues for future research and advancement.

The References section lists all the sources cited in this document, ensuring comprehensive documentation and offering possibilities for additional reading and verification.

2. State of the Art

This chapter presents a review of existing literature and techniques concerning the automated identification, diagnosis, and classification of diseases and pests in agricultural crops. This chapter's main aim is to offer a comprehensive understanding of the most recent advancements and methodologies in this field, which will serve as a foundation for the suggested system. By reviewing the existing literature, the objective was to highlight the progress and techniques that have influenced this topic. These findings will guide the development of the system, ensuring that it is constructed using the most efficient strategies now accessible.

The research presented in this chapter contributed to the writing and publication of a scientific article, which may be accessed at [7]. This article explores the application of machine learning in automatically detecting diseases in agricultural crops through the analysis of leaf photographs. The paper examines different machine learning techniques and their use in disease detection and classification, establishing a strong basis for the suggested approach. The comprehensive examination in the following parts is a direct outcome of this thorough investigation, culminating in the dissemination of a scientific study. This study emphasizes the thoroughness and meticulousness of our analysis, ensuring that our system development is grounded in well documented and peer-reviewed research.

In addition, an analysis was conducted on several existing systems that utilize machine learning to automatically detect diseases in agricultural crops using leaf photos. The apps listed, such as Plantix, PictureThis, Leaf Doctor, and Agronomist.ai, demonstrate the practical application and efficacy of the aforementioned approaches. Through the examination of these applications, it is possible to obtain valuable insights into the practical implementation of these technologies, which will greatly help to the progress of the proposed system. Each of these programs uses advanced algorithms to deliver accurate and quick diagnoses, assisting farmers in efficiently controlling the health of their crops.

This chapter establishes the foundation to define the requirements and design a system that combines research findings and practical uses to tackle the difficulties of identifying diseases in agricultural crops.

2.1 Purpose of the Review

An extensive review was undertaken to understand the automated procedures involved in the detection and classification of diseases and pests in agricultural crops. Information regarding the type of algorithms, models, and their characteristics is crucial to understand the various approaches' efficiency and applicability. More specifically, the data gathered in this review is used to answer the following research questions:

RQ1 - Which crops are most focused on studies of automatic detection of diseases and pests?

RQ2 - Do the studies address more diseases or pests?

RQ3 - What inputs were most used in algorithms to detect and classify diseases and pests?

RQ4 - What are the most used datasets for these studies?

RQ5 - What are the most used algorithms and models?

RQ6 - What levels of accuracy have been achieved in the automatic identification and classification of diseases and pests in agricultural crops?

RQ7 - What trends have been most noticed in this area of study?

The answers to these questions will help characterize the state-of-the-art and the main trends in this study area. The remainder of this chapter is organized as follows. Section 2.2 details the related work in the field, providing a comparison with previous studies. Section 2.3 describes and applies the chosen methodology to perform this review, presenting the search strategy, the inclusion and exclusion criteria, and the results summary. Data extraction and analysis are elaborated upon in Section 2.4. The findings are previously discussed in Section 2.5. Section 2.6 highlights the strengths and limitations of this review. Section 2.7 explores related applications that use machine learning for disease detection in agriculture. Finally, Section 2.8 provides concluding remarks, summarizing the insights and implications of the review.

2.2 Related Work

In recent years, several studies have reviewed works related to the automatic detection, identification, and classification of diseases and pests in agricultural crops. To better understand the work done in this area, a search was done in the Scopus database. Thus, after some initial experiments and considering an initial analysis of some literature studies, five terms were identified and used in the search. First, terms related to the type of study, in this case, review. Second, the term automatic. Third, terms related to pests, plagues, or diseases. Fourth, terms related to agriculture, crops, leaves, or plants were added to the search terms, as they represent the study area or plants or parts where diseases or pests are typically visible. Finally, classification, identification, or detection terms were also included.

The complete string for the search is as follows:

review AND automatic AND (pest OR plague* OR disease*) AND (crop* OR leaf* OR plant* OR agricul*) AND (classification OR identification OR detection)*

The database search was conducted in February 2023 and considered the field's article title, abstract, and keywords. Only studies published in 2012 or later were considered. After searching for the literature, 98 studies were obtained. These studies were evaluated in terms of title and abstract, resulting in the exclusion of 78 studies. Most of them were excluded because they were not studies presenting reviews. After all, their focus differed from the intended one (5) or because their full text was unavailable (3). The remaining 20 studies were analyzed in qualitative and quantitative analysis. Table 1 summarizes some of their characteristics, namely the year in which the article was published, the main objective of the review, the crop(s) analyzed in the review, the number of studies included in its analysis (when the analysis covers several areas, only the studies that are focus on identification or classification of diseases or pests are considered), and the time span of studies considered in the review.

Table 2. Studies reviewing the detection, identification, and classification of diseases and pests in agricultural crops.

Review	Year	Focus	Crop	Studies	Timespan
[8]	2022	Artificial Intelligence-based disease diagnosis and categorization techniques for paddy crops.	Rice	20	2016–2021
[9]	2022	Rice leaf diseases based on the Deep Convolutional Neural Network (D-CNN) architectures	Rice	8	2019-2020
[10]	2022	Machine Learning (ML) based techniques for forecasting, detection, and classification of diseases and pests, with an emphasis on tomato crops	Tomato	16	2006-2020
[11]	2022	Computational methods operated in different stages of plant-pathogen systems like image preprocessing, segmentation, feature extraction and selection, and classification to diagnose the diseases	Cotton	6	2012-2018
[12]	2022	Review of 100 Convolutional Neural Network (CNN) approaches to detecting plant leaf diseases.	Multiple	100	2015-2022
[13]	2022	Tomato leaf disease identification using image processing, ML, and deep learning approaches	Tomato	44	2015-2022
[14]	2022	Plant leaf disease recognition and classification using deep learning and advanced imaging techniques	Multiple	50	NA
[15]	2022	ML and deep learning for maize leaf disease classification	Maize	62	2015-2022
[16]	2022	Diseases that harm potato crops and a survey on computer vision-based techniques and types of ML algorithms used.	Potato	39	2016-2022
[17]	2022	Crop disease and pest recognition using image processing to extract the features and algorithms used in prediction studies	Multiple	10	2015-2020
[18]	2022	Plant disease detection using ML	Multiple	15	2014-2021
[19]	2022	ML and Deep Learning based approaches for plant leaf disease classification:	Multiple	30	1970-2022

[20]	2022	Image processing techniques with the help of artificial neural networks for automatic pest detection	Multiple	16	2012-2021
[21]	2021	Trends and challenges for the detection of plant leaf disease using deep learning and advanced imaging techniques	Multiple	42	2019-2020
[22]	2021	D-CNN approaches for identification and diagnosis of various diseases on plant leaves	Multiple	7	2016-2019
[23]	2021	Image processing techniques employed in pest and disease recognition using visible light (RGB) images\	Multiple	51	2015-2020
[24]	2020	ML technologies for plant diseases detection	Multiple	25	2016-2022
[25]	2020	Grains disease detection, quantification, and classification using computational techniques	Grains	109	2001-2020
[26]	2020	Deep learning techniques in plant disease detection	Multiple	18	2015-2018
[27]	2020	Sensors for automatic detection and monitoring of insect pests	Multiple	63	2004-2020

It is evident the interest that the automatic detection, identification, and classification of diseases and pests in agricultural crops have attracted and the growing importance it has gained in recent years. After applying the inclusion and exclusion criteria, all identified studies were published between 2020-2022, with a higher prevalence in 2022. Furthermore, more than two-thirds of the studies only analyzed articles published in the last ten years (published in 2013 or later). This highlights the interest that this study area has attracted in recent years. It also indicates that it is still a new area of research and has been the focus of more significant interest only recently.

ML approaches are very popular for automatically detecting and monitoring diseases and pests. Eighty percent of the reviews specifically address the use of ML algorithms and, more specifically, deep learning algorithms.

Regarding the type of crops studied, 40% of the articles describe reviews of studies that specifically addressed a single crop (e.g., tomato, rice, cotton, potato). The remaining 12 (60%) reviews analyze studies related to multiple crops.

The number of studies analyzed in each review is quite diverse. Twenty percent of reviews analyze ten studies or fewer, and 45% analyze 20 studies or fewer. Considering the number of studies that have appeared in recent years, the number of studies included in these reviews is insignificant. On the other hand, 40% of the reviews analyze 40 or more studies. Three reviews analyze specific crops (tomato, maize, and grains). The other five reviews analyze multiple crops. One of them ([27]) presents a review of sensors for the automatic detection and monitoring of insect pests, which is different from the focus of the review presented in this article. Considering the dimension of the review, i.e., the number of studies included in each review and the scope, the remaining four reviews ([12][14][21][23]) have some similarities with the review presented in this article. However, some of them focus on approaches that use specific algorithms, [12] focus on CNN in the detection of plant leaf diseases, and [14][21] focus on deep learning-based plant disease detection. Some differences focus on the parameters considered in the analysis and the research questions investigated. In this sense, this work represents a step forward concerning other related works, thus representing a significant contribution to this study area.

2.3 Methodology

This section reviews studies that addressed automatic detection and identification systems for diseases and pests in agricultural crops. The review was done following the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) statement [28]. It includes the following steps, which correspond to the section or subsection of this article listed after each step:

- Identifying the purpose and intended goals of the review (Section 1).
- Search strategy (Section 2.3.1).
- Screening for inclusion (Section 2.3.2).
- Screening for exclusion (Section 2.3.3).
- Data extraction (Section 2.4).
- Analysis and Discussion (Sections 2.5).
- Writing the review.

Next, the procedure used to arrive at the related work, the data sources and keywords used in the research, the process, data selection, data extraction, and analysis, which will be used in this work, are described.

For the analysis of the related work, articles that address the use of algorithms for automatically identifying diseases and pests in crops were studied. After obtaining the first results, the data underwent a selection process to eliminate irrelevant articles or articles that were out of context. After selecting the relevant articles, they were analyzed according to predefined parameters.

2.3.1 Search Strategy

The Scopus [29] and Web of Science [30] databases were used as data sources. These databases are among the most complete in several areas and provide an advanced search that allows users to configure search words in different fields, such as in the title, keywords, and throughout the text, among others. It also allows adding logical operators: AND, OR, and NOT. In this way, access was gained to a significant part of scientific work in informatics.

In collecting the first sample of articles, the keyword "automatic" and terms related to diseases or pests ("pest," "disease," plague), terms related to crops (crop, leaf, plant or agriculture), and terms related to identification (classification, identification, and detection) were used. To make it possible to search for articles that contain the indicated terms, the symbol "*" was used to represent other terms that may contain the identified keywords. Thus, the string for the search is as follows:

automatic AND (pest OR plague* OR disease*) AND (crop* OR leaf* OR plant* OR agricul*) AND (classification OR identification OR detection)*

2.3.2 Screening for Inclusion

In this screening, the reviewers considered that studies should only be included in the review if they met the following criteria:

- (1) Studies that presented a solution for automatic detection and identification systems for diseases and pests in crops;
- (2) Studies with full text available.

Studies that met only some of these criteria were excluded. After removing duplicated articles (23), 255 studies remain. Moreover, after applying criteria (1), 143 more records were excluded. At this stage, the reviewers did not judge the quality or

evaluate the information found in each study. Furthermore, after applying criteria (2), more than 29 were excluded. It results in 83 studies.

2.3.3 Screening for Exclusion

With a more in-depth reading of the 83 articles, 35 were eliminated for needing more information or being out of the intended focus (reviews (6) and incomplete info (29), leaving 48 articles to analyze.

2.3.4 Results Summary

As shown in Figure 1, after searching the literature, 255 papers were obtained (after removing 23 duplicates), referred to as the 'identification' stage in the diagram; after applying the inclusion criteria identified in Section 3.2 'Screening for Inclusion' and in the 'screening' section of the diagram, 172 papers were excluded, resulting in 83 papers.

A full-text evaluation of the papers was performed, thus excluding papers that did not match the intended focus, some that were just a review, and papers that did not have complete information; this step is represented in the Figure as "eligibility."

The 48 papers remaining at the end were featured in the synthesis and were the "included" studies in the flowchart.

This study focused on finding papers directly linked to identifying, detecting, and eventually automatically classifying anomalies in agricultural crops. In this context, the following papers show several examples of this process where several techniques were presented as a solution. In addition, some datasets were mentioned, where one can see the origin of the information in each experience presented. Also, taking into account that one of the main focuses was to highlight the accuracy of the algorithms or models used to classify diseases or pests in crops.

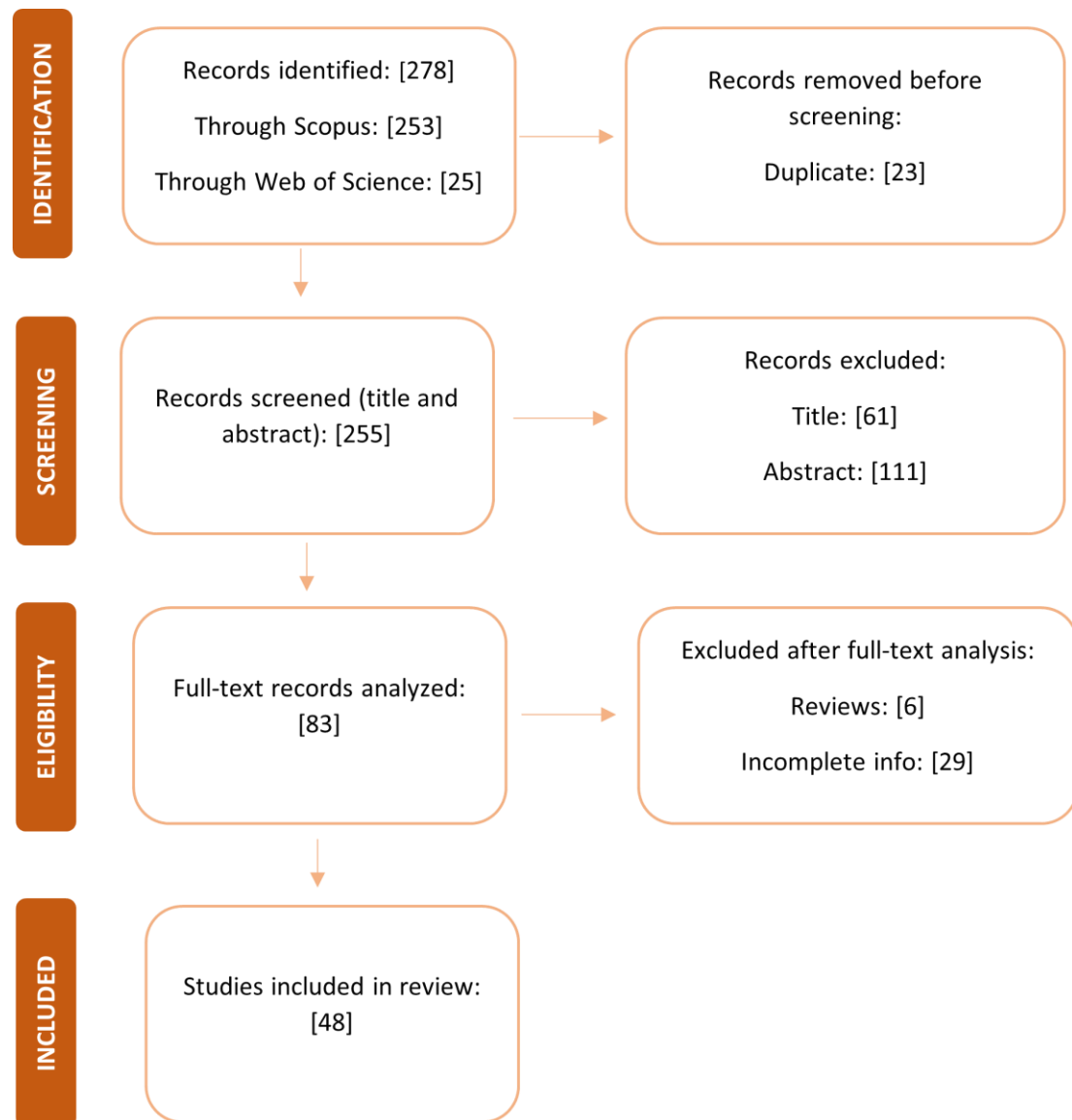


Figure 1. The flowchart of the systematic review process determined the papers selected for the present study (adapted from the PRISMA 2020 flow diagram [31]).

2.4 Data Extraction and Analysis

Each of the 48 selected articles was analyzed and described considering the following criteria:

- Year of publication.
- What types of crops is it intended for?
- What algorithms are used?
- What are the inputs for the proposed algorithms?
- What information is used for the training and validation of the algorithms?
- What results were obtained in terms of accuracy and diseases or pests identified?

Next, the analysis of the 48 articles is presented, considering these perspectives. A comparative table of the analyzed articles is presented at the end of the analysis, considering the different analysis fields.

In [32] is described as a system that uses computer vision and deep learning methods to detect and count common pests in an outdoor experimental environment. An integrated platform is presented as an information collector for obtaining the real-time status of a sticky trap. A remote server is used for data processing and management. A detection network using a self-designed backbone DPeNet was applied for the detection and counting tasks. The DPeNet successfully detected pests of different sizes. The system uses dedicated hardware to capture images of insects, and the algorithm uses, as input for training, a dedicated dataset of images.

A citrus pest' detection using deep learning methods is proposed in [33]. High-resolution citrus pests of fruit and leaf images of natural plantations were collected to construct a multivariate dataset of citrus images. One of the main focuses was that the citrus pest pattern was displayed in various conditions. They use models such as VGGNet, ResNet, DenseNet, EfficientNet, and Vit (passed through transfer learning), which facilitated the process in the dataset training for detecting citrus diseases. All models used in the experiment had an average accuracy of 97% or more and an average f1 score of 96% or more.

In [34], the authors presented a system for pest detection in granaries. It consists of two main parts: a deep learning object detection model called YOLOv4 and a small car with a camera and an auxiliary light. This model was trained and embedded in the car, and the camera on the small car can take photos of the pests for YOLOv4 to identify the species and number of pests. In conclusion, it is claimed that the robotic system can regularly take photos of the granary surface without dead ends. Moreover, during this process, pest information can be accurately obtained through the object detection method YOLOv4.

A deep learning approach is proposed to automate identifying and classifying diseases affecting different parts of the rice plant, described in [35]. Models such as VGG-16 and GoogleNet CNN pre-trained on ImageNet were used, and tested for 24 rice disease symptom images containing 500 images of each symptom, providing a labeled dataset of 12,000 images. The efficiency of the models considered is improved using the threefold cross-validation method. Overall average classification accuracies obtained using VGG-16 and GoogleNet CNN models are 92.24% and 91.28%, respectively. Comparing the results of both the CNN models, the performance of the pre-trained VGG-16 CNN model shows slight improvement over the pre-trained GoogleNet CNN model in the classification of rice diseases.

In [36], a deep learning model was proposed to help early detection of diseases caused by begomoviruses. Specifically, a deep learning architecture based on a CNN known as Visual Geometry Group 16 or VGG-16 was employed to detect begomovirus infections in tobacco plants inoculated with ToLCNDV and ToLCGV. To obtain a more generalized result, the dataset used by the authors included leaves from plants at different stages of viral infection. The entire dataset was divided into two classes, healthy and unhealthy. The output was an efficient CNN model with an accuracy of 97.211%.

In [37], the authors described an approach to disease identification, discovering the differences in the morphological characteristics existing in the patterns of image texture features and multi-trait textures of cassava plants. According to them, a broad database of images of the cassava plant would be required to carry out the same, which was classified by the proposed classification system based on a distributed structure based on the revelation of hidden patterns existing in the texture resources of the image. Diseases were classified by a distributed structured k-NN classifier employed in

the binary approach. The system achieved satisfactory accuracy rates for all classes considering the dataset contains images captured in natural conditions with complex backgrounds.

In [38], the authors proposed a Tranvolution detection network with GAN modules for plant disease detection. Also, a generative model has been added to the front of the backbone, and GAN models have been added to the attention extraction module to build GAN modules. The dataset used in this study, PlantDoc, published by researchers at the Indian Institute of Technology, is a collection of 13 plant species and 27 categories (including 17 types of diseases). The experiments showed that the proposed method satisfactorily reached 51.7% (Precision), 48.1% (Recall), and 50.3% (mAP), respectively. Also, the SAGAN model was the best in the attention-extraction module, while the WGAN model had the best performance in image enhancement. The proposed model was implemented in the Hbird E203, and they created an intelligent agricultural robot to put the model into practical agricultural use.

An automatic classification technique and early detection of diseases for aromatic plants using hybrid flexible computing techniques (CDD-H-HSC) are described in [39]. In the first instance, a multi-swarm coyote optimization (MSCO) algorithm is used for disease segmentation that separates the disease area from the input plant leaf image. Second, the Improved Chan-Vese Snake Optimization Algorithm (ICVSO) is applied to optimize the features to select the best optimal among multiple features, which reduces the dimensionality problem. After that, the authors developed a hybrid flexible computing technique, which works as a deep neural network classifier (FDB-DNN) of distance and aptitude balance, classifies the leaf, and detects the disease. Results show that the accuracy of the proposed FDB-DNN classifier is 2.8140%, 1.507%, 5.125%, and 0.402% higher than the existing state-of-art SVM, PNN, NB, and SVM+PNN+NB+FIS classifiers respectively.

In the study described in [40], the authors present the results obtained in classifying foliar diseases in sunflowers using a system composed of a model that automatically segments foliar lesions, followed by a classification system. Lesion segmentation was performed using Faster R-CNN and Mask R-CNN. The residual neural networks ResNet50 and ResNet152 were used for disease classification based on lesions. They claim that the results show that automatic segmentation of lesions can be successfully achieved in the case of diseases such as Alternaria and Rust, in which the lesions are

well delineated. They conclude that using a system composed of a network that segments the lesions, followed by a network that classifies the diseases, allows one to classify the diseases more precisely and to identify those images for which a precise classification cannot be made. Results of the segmentation-classification system for Faster R-CNN and the segmentation-classification system for Mask R-CNN vary from 31% of the minimum precision value and 96% maximum.

In [41], the authors describe an integrated approach that uses the MIB classifier. The MIB classifier model differentiated unhealthy and healthy fruits or leaves with similar citrus diseases. They used the Gaussian feature extraction method to extract the layers of various images. Furthermore, the trained deep learning model was verified on several citrus fruit/leaf datasets. Moreover, as for the proposed results, the authors demonstrate that the MIB classifier model performs better than the challengers in several counts of computational data. The model trained with the MIB classifier and Gaussian feature extraction offers the best accuracy of 98%.

A five-layer CNN model for automatically detecting plant diseases using leaf images is presented in [42]. To train a CNN model, 20,000 augmented images are generated. As for classifying plant foliar diseases, they used the pepper leaf dataset consisting of 2,475 leaf images. They demonstrate that the proposed optimized CNN model can predict bell pepper leaves as healthy or bacterial with 99.99% accuracy. In addition, they show that the highest accuracy of 99.99% was achieved with 25 epochs.

In a study described in [43], the authors proposed an automatic disease detection system that recognizes the category of diseases and the healthiness of plants in several crops. They studied a way of classifying diseases through symptoms that appeared on the leaf's surface. Their study broadly categorizes diseases into three categories. CNN is used for disease classification. The dataset consists of 64,963 images, where training was performed on 80% of the dataset samples, and validation was performed on 20% of the dataset samples. Simulation results show that the trained CNN achieved an accuracy of 99.12%.

To address the problem of detecting and recognizing wild grassland pests, a study [44] proposed an automatic data augmentation method to adaptively search for the appropriate augmentation strategy and model the data more effectively. They used Res2Net as a backbone to get richer detailed information about tiny pests, and a reverse feature fusion layer was introduced into Feature Pyramid Networks (FPN) to

learn more details. They also evaluated the performance of their method on the pest dataset, including 4k and 4-class images. Their method achieves pest detection performance of 81.0% average accuracy (mAP), which improves 5.7%, 4.0%, and 3.1% compared to three state-of-the-art approaches YOLOv4, Faster R-CNN, and Cascade R-CNN detectors, respectively.

A study [45] analyzed the performance of pre-trained CNN models VGG16 and VGG19 to classify diseases of various plants like basil, bael, Arjun, etc. There are 21 classes in the dataset. The training and testing performed with all models were 80% of the dataset for training and 20% for validation purposes. The validation accuracy of VGG16 yielded the best results at 98.52%, whereas VGG19 showed an accuracy of 98.08%. Data augmentation was applied to reduce the problem of overfitting and generalization.

In [46], the authors proposed a model based on a CNN with 19 convolutional layers to effectively and accurately classify Marsonina Coronaria and Apple Scab diseases of apple leaves. They acquired a database of 50,000 images by collecting leaf images from apple farms in Himachal Pradesh (H.P) and Uttarakhand (India). A magnification technique was performed on the dataset to increase the number of images to increase accuracy. The performance analysis of the proposed model was compared with the two CNN models with 8 and 9 layers, respectively. The proposed model was also compared with standard ML classifiers such as support vector machine models, KNN, random forest, and logistic regression. Experimental results showed that the proposed model outperformed other CNN-based and ML models with an accuracy of 99.2%.

In [47], the authors describe a framework for detecting the disease on brinjal leaves. It comprises five stages: image preprocessing, segmentation, fusion, feature extraction, and classification. One of the main intentions of the authors was to show that the lesion areas affected by diseases are segmented. In addition, an image fusion technique based on discrete wavelet transform is used to improve image quality reduction redundancy. The merged images' color, texture, and structural features are extracted and provided as input to the ANN for classification that provides better performance. The overall rating rate using FFNN is 91%, and for CFNN, it is 98.1%.

In [48], the authors describe the Location-wise Soft Attention engine to the pre-trained MobileNet-V2, in which the general knowledge of images learned from ImageNet was migrated to their disease recognition mode of cultures. A localization

strategy was adopted, where the Disease Activation Map exhibited the significant regions of the images that caused the classification. Notably, the transfer learning was conducted twice in model training, which gained the optimal weight parameters for the model. In their experiments, they achieved an average accuracy of 99.71% on the open-source dataset, and even under disordered background conditions, the average accuracy reached 99.13% for identifying disease in crops.

In [49], a transfer learning model (DenseNet201) was designed by the authors to conduct the recognition of symptoms of three tomato diseases, such as leaf mold, late blight, and late blight. The transfer learning model was trained, tested, and validated. Their trained model can recognize the disease category. Once the images are classified, regions are segmented, and features are extracted using k-means clustering and ORB key points. Their model achieved a recognition result with an average accuracy of 99.688%.

In another study [50], the authors proposed using a deep learning architecture based on a recent CNN called EfficientNet on 18.161 images of simple and segmented tomato leaves to classify tomato diseases. The performance of two segmentation models, namely U-net and modified U-net, for segmenting leaves is reported. Also reported are the comparative performance of binary classification (healthy and unhealthy leaves), six-class classification (healthy leaves and various groups of diseased leaves), and ten-class classification (healthy leaves and various types of unhealthy leaves) models. The modified U-net segmentation model showed an accuracy of 98.66% for the segmentation of leaf images. According to their results, EfficientNet-B7 showed superior performance for binary classification and six-class classification using segmented images with an accuracy of 99.95% and 99.12%, respectively. Finally, EfficientNet-B4 achieved an accuracy of 99.89% for classifying ten classes using segmented images.

A whitefly and thrips detection model based on images of sticky traps acquired in a greenhouse as described in [51]. An end-to-end model based on Faster R-CNN, named TPest-RCNN, was developed to improve the detection accuracy of tiny pests. This architecture was trained using a TL strategy on the Common Objects in Context dataset before training on the small pest training set to create the TPest-RCNN model. The new model achieved a mean F1 score and mean precision of 94.4% and 95.2%, respectively, in a validation set.

An approach to detecting and classifying tea leaf diseases, incorporating deep learning techniques, was described in [52]. In this work, they applied the CNN model with one input layer, four convolution layers, and two fully connected layers. The crux of this work was to classify and improve the accuracy of correctly predicting diseases. The proposed model considers several tea leaf diseases and achieves correct identification with 94.45% accuracy.

In [53], the authors present a fine-tuned model and perform its performance comparison with VGG16 using the PlantPathology data set. The experimental results show how the proposed model performs better than VGG16. The model also exploits the transfer learning approach's advantages and freezes selected layers of the VGG16 CNN model to achieve better accuracy. The experimental results show the performance of the fine-tuned model while classifying apple leaves into three categories: healthy, scab, and cedar apple rust. After a series of tests, the model yielded an accuracy of 85% while classifying the leaves images into three classes.

A study [54] presents an automated approach to pest detection using Wavelet transformation. The main objective of the research is to improve the feature extraction phase to improve detection efficiency. Furthermore, as for results regarding the algorithm's accuracy, they claim that the proposed approach is invariant with color and rotation factors. The efficiency of the presented approach is 91.89%, and the recall and precision are 0.92 and 0.96, respectively.

In [55], the authors proposed a system that, based on visible light spectrum images of its leaves, allows the detection of conditions in citrus crops. The system is designed with deep learning and image processing: CNN was trained using transfer learning and data augmentation, among other techniques. For diagnosis, the affections are classified into seven categories. The main findings include accuracy rates of more than 90% when using the classification.

In [56], the authors proposed a technique based on deep learning for classifying citrus diseases. Two different pre-trained deep learning models were used in this work. Image enhancement techniques are used to increase the size of the citrus data set in this article. In addition, hybrid contrast stretching was adopted to improve the visual quality of the images. Transfer learning is used to retrain the pre-trained models, and the feature set is enriched using feature fusion. Finally, they also state that the selected

features are used to classify six different diseases of citrus plants. The proposed technique achieves a classification accuracy of 95.7%.

In [57], the authors proposed an approach based on a CNN model to differentiate healthy fruits and leaves from fruits/leaves with common citrus diseases. They compare their approach against many state-of-the-art deep learning models in the Citrus and PlantVillage datasets. Regarding accuracy, their experimental results indicate that the CNN model outperforms competitors in various measurement metrics. Their CNN model has a test accuracy of 94.55%.

The study published in [58] describes and evaluates several methods for identifying plant diseases with little data. CNNs are used due to their superior ability to transfer learning. For this purpose, three CNN architectures (ResNet18, ResNet34, and ResNet50) were used to build two baseline models, a Triplet network and a deep adversary metric learning (DAML) approach. Finally, the evaluation results demonstrated that a baseline model trained with a large set of source field images could be adapted to classify new diseases from a small number of images. They were also able to take advantage of the availability of a greater number of images. Furthermore, comparing it with metric learning methods, the authors could see that the baseline model has better transferability when the source domain images differ significantly from the target domain images or are captured under different conditions. It achieved an accuracy of 99% when the change from the source domain to the target domain was small and 81% when the change was significant.

In [59], image segmentation is described into more than 450 images of healthy and diseased potato leaves taken from the publicly available plant village database. Seven classification algorithms are used to recognize and classify diseased and healthy leaves. Among the classifiers used for the accuracy test, the Random Forest classifier gives an accuracy of 97%.

In [60], 54,303 labeled images from the PlantVillage dataset were used, comprising 38 plant leaves and disease types (classes). Data were divided into the source (32 classes) and target (6 classes) domains. The Inception V3 network was tuned in the source domain to learn the general characteristics of the plant leaf. Furthermore, according to them, the algorithm's performance was evaluated through the total accuracy, precision, and recall per class. For the FSL experiments, the algorithms were trained with different numbers of images per class, and the experiments were repeated

20 times to characterize the results statistically. In terms of results, they can see that the accuracy in the source domain was 91.4% (32 classes), with a median precision/revocation per class of 93.8%/92.6%. The accuracy in the target domain was 94.0% (6 classes) learning with all training data, and the median accuracy (90% confidence interval) learning with one image per class was 55.5% (46.0% –61.7%).

A model that assists in automatically detecting different tomato diseases in their early stages was proposed in [61]. The main objective was to identify different types of foliar diseases. Different feature extraction techniques have been used to improve classification accuracy. Support Vector Machine (SVM), Random Forest, and Logistic Regression have been applied to classify different types of foliar diseases. The results show that the model can be used in real-life applications.

Some authors propose a system [62] to develop image-processing techniques for detecting and alerting diseases. The steps involved are image acquisition, image processing, image segmentation, feature extraction, disease classification, and categorization. After detecting the disease, it sends alerts through the buzzer. Leaf detection and alerting are done using the k-means clustering algorithm in two methods, particularly in GUI and real-time. In GUI, the results have an accuracy of 95.1613% for *Alternaria Alternata* disease. In real-time, when the disease is detected, it warns people through a buzzer, and when the healthy leaf is detected, it does not turn on the buzzer and remains silent.

In [63], the authors studied the transfer learning of the D-CNN for identifying plant foliar diseases. They considered using the pre-trained model learned from the typical massive datasets, and then they transferred to the specifically trained task with their data. VGGNet pre-trained in the ImageNet and Inception module is selected in their approach. Instead of starting the training from scratch by randomly initializing the weights, they initialized the weights using the pre-trained networks on the large-labeled dataset, ImageNet. Moreover, even under complex background conditions, the average accuracy of the proposed approach reaches 92.00% for the prediction of rice plant image classes.

In [64], the authors present an analysis of network weights for automatic recognition of soybean leaf diseases applied to images taken directly from a small unmanned aerial vehicle (UAV). To obtain high accuracy, they evaluated four models of D-CNN trained with different parameters for fine-tuning and transfer learning. In the

classification step, they compared four deep learning models well-known in the literature: Inception-v3, Resnet-50, VGG-19, and Xception. Experimental results showed that deep learning models lead to high classification rates, reaching an accuracy of up to 99.04%. Furthermore, in their experiments, the 100% and 75% fine-tuning strategies showed higher classification rates compared to other training strategies.

An algorithm called Enhanced Fusion Fractal Texture Analysis (EFFTA) was proposed in [65]. The proposed method consists of the feature fusion technique that combines Scale Invariant Feature Transform (SIFT) and Discrete Wavelet Transform based on Segment-Based Fractal Texture Analysis. The image can be detected by shape, texture, and color. Finally, they carried out a texture feature and selection method that combines SIFT and WSFTA extraction techniques and selects the best feature by running the Principal Component Analysis (PCA) algorithm for further classification. Their tests show that the proposed method produces the best result of 97.69% when the extracted features are classified using BPNN.

In [66], an updated Faster R-CNN architecture developed by changing the parameters of a CNN model and a Faster R-CNN architecture for automatic detection of leaf spot disease (*Cercospora beticola* Sacc.) in sugar beet were proposed by the authors. The method proposed for disease severity detection by image-based expert systems was trained and tested with 155 images, and according to the test results, the overall correct classification rate was 95.48%.

In [67], the authors present a proposal for detecting diseases in fruits using their physical attributes. Their proposed methods comprise the K-Means clustering technique, ANN, and SVM. K-Means is used for image segmentation. It maps the images to the respective classes of diseases based on phenotypic characteristics such as texture, color, the structure of holes in the fruit, and physical constitution. The classifiers are trained and tested utilizing images of citrus fruit diseases. The color and texture highlights are utilized for training and testing the Artificial neural network model. The accuracy rate of recognition and classification is characterized as the proportion of accurately recognized sample images to the total number of sample images chosen.

A D-CNN model to classify insect species in three publicly available insect datasets was proposed in [68]. They used three datasets. The National Bureau of Agricultural

Insect Resources (NBAIR) dataset is the first insect dataset, including 40 field crop insect images. In contrast, the second and third datasets (Xie1, Xie2) contain 24 and 40 insect classes, respectively. The proposed model was evaluated and compared with pre-trained deep learning architectures such as AlexNet, ResNet, GoogLeNet, and VGGNet for insect classification. They achieved a higher classification accuracy of 96.75%, 97.47%, and 95.97% in the proposed CNN model for the NBAIR insect dataset (40 classes), Xie1 insect dataset (24 classes), and Xie2 insect dataset (40 classes), respectively.

In [69], a mobile application model based on deep CNN to recognize tomato leaf diseases is proposed. For the development of this application, their model was inspired by the MobileNet CNN model. It can recognize the ten most common types of tomato foliar diseases. Trained on the tomato leaf dataset, 7176 tomato leaf images were used in the smart mobile system to perform a tomato disease diagnosis. Their adapted CNN model was tested with many optimization algorithms such as gradient descent stochastic, adadelata optimizer, adagrad, adagradDA, Momentum, Adam, Ftrl, proximaladagrad and RMSprop optimizers. Furthermore, their adapted model was also tested with many training steps and changing many configuration parameters.

In this paper [70], the authors describe a deep convolution neural network (D-CNN) based approach to identify guava leaf diseases automatically. The model they propose can be applied to classify the primary guava leaf diseases. They created their dataset called BU_Guava_Leaf with four different categories. On this dataset, they train their proposed disease identification approach and evaluate the experimental result, which shows an average accuracy of 98.74% in the test set.

A system using an image processing approach for automatically diagnosing cotton foliar diseases were described in [71]. Classification is based on selecting appropriate features, such as image color and texture, using the SVM classifier. The images are acquired by themselves from cotton fields using a digital camera. Various preprocessing techniques, such as filtering, background removal, and enhancement, were used. Finally, the composite classifier works correctly and with recognizable evidence of disease, giving an accuracy of 98.46%.

In [72], the authors proposed an automatic disease detection system in grapevines using image processing and ML techniques. Using the capture slice segmentation method, the system segments the sheet (Region of Interest) from the background

image. The features are extracted from the segmented diseased part, and it was classified as healthy, rot, esca, and rust using different ML techniques such as SVM, adaboost, and Random Forest tree. Using SVM, they were able to achieve a better test accuracy of 93%.

In a study described in [73], the authors adopted a slight variation of the CNN model called LeNet to detect and identify diseases in tomato leaves. Neural network models use automatic feature extraction to help classify the input image into respective disease classes. They achieved an average accuracy of 94%-95%.

A system to automatically detect two defoliating pests of potato and bean crops: Mexican Bean Beetle (MBB) and Colorado Potato Beetle in adulthood, was described in [74]. The neural classifiers used for beetle detection are Random Subspace Classifier (RSC) and limited Receive Area (LiRA). The MBB images used as inputs for the classifiers were obtained in Mexican plantations. They compared the results obtained with both classifiers on image databases. The RSC classifier has the best recognition result, 89%, while the LiRA has a recognition rate of 88%.

In [75], the authors explore detection algorithms on leaf images and build a convolution neural network model to detect tomato pests and diseases based on VGG16 and transfer learning. The detection model is trained with the Keras/TensorFlow deep learning framework and, according to them, achieved an average classification accuracy of 89%.

In [76], a method for identifying rice diseases based on D-CNN techniques is presented. Using a dataset of 500 images of diseased and healthy rice leaves and stems captured in the experimental rice field, CNNs are trained to identify ten common rice diseases. Under the 10-fold cross-validation strategy, the proposed CNN-based model achieves an accuracy of 95.48%.

Another study [77] presents an algorithm for an image segmentation technique for automatically detecting and classifying plant foliar diseases. The study also covers the survey of different disease classification techniques that can be used to detect plant foliar diseases. The classification is done first using the Minimum Distance Criterion with K-Mean Clustering with an accuracy of 86.54%. The detection accuracy is improved to 93.63% by the proposed algorithm. In the second phase, the classification is done using the SVM classifier and shows its efficiency with an accuracy of 95.71%.

In [78], the authors present a combination of thermal light and visible light image data with depth information and an ML system to detect plants infected with the tomato powdery mildew fungus *Oidium neolycopersici*. They extracted features from the image data using local and global statistics. They showed that combining them with the depth information could considerably improve diseased plants' accuracy detection. Their results show that, by combining information from stereo and thermal visible light imaging and using ML techniques, tomato plants infected with *Oidium neolycopersici* can be identified with high accuracy - over 90%.

In the study described in [79], the authors use a traditional ML method to gather the output of a deep learning model to identify peanut leaf diseases. Identifying peanut leaf diseases included healthy leaves, single leaf blight, single leaf spot, single leaf blight, and single leaf blight and blight. Moreover, three types of data augmentation methods were used in their research: image flipping, rotation, and scaling. In this experiment, the deep learning model was more accurate than traditional ML methods. Also, the deep learning model performed better when using data augmentation and a pushdown set. After ensemble by logistic regression, the accuracy of the residual network with 50 layers (ResNet50) was 97.59%, and the F1 score of the dense convolutional network with 121 layers (DenseNet121) was 90.50.

A summary of the characteristics of the included articles is summarized in Table 3.

Table 3. Summary of the articles selected for analysis and synthesis.

Paper	Year	Application	Disease or Pest	Crop	Disease or Pest Type Found	Input
[32]	2022	Web & Mobile	Pest	N/A	N/A	Insect images from the sticky trap
[33]	2022	Web	Pest	Citrus	Citrus bacterial canker (CBC), <i>Toxoptera citricida</i> & <i>Panonychus citri</i>	Fruit and leaf image
[34]	2022	Robotic System	Pest	N/A	Red Flour Beetle and Rice Weevil	Leaf images from a small car with a camera and a supplementary light
[35]	2022	Algorithm	Disease	Rice	Grassy stunt virus, tungro virus, and yellowing mottle	Plant images
[36]	2022	Algorithm	Disease	Tomato	ToLCNDV & ToLCGV (begomovirus infections)	Leaf images
[37]	2022	Algorithm	Disease	Cassava	Bacterial blight, Brown streak disease	Leaf imagens
[38]	2022	Robotic System	Disease	N/A	17 types of diseases (Bell Pepper Bacterial, Apple Black Rot, and others)	Leaf images
[39]	2022	Algorithm	Disease	Herbs	Gray mold, rust, and scab disease	Leaf images
[40]	2022	Algorithm	Disease	Sunflower	Alternaria, Rust	Leaf images
[41]	2022	Algorithm	Disease	Citrus	Powdery mildew, downy mildew	Leaf images
[42]	2022	Algorithm	Disease	Pepper	Black spot, Melanose, Scab, Greening, Canker	Leaf Image
					Pepper bell leaf bacterial	Leaf images

[43]	2022	Algorithm	Disease	Apple Corn Grape Peach Pepper Potato Strawberry Tomato	Fungal, bacterial, and viral	Leaf images
[44]	2022	Algorithm	Pest	Maize	Wheat mites, wheat aphids, wheat sawflies, rice plant hopper	Leaf images
[45]	2021	Algorithm	Disease	N/A	N/A	Plant images
[46]	2022	Algorithm	Disease	Apple	Marsonina Coronaria, Apple Scab	Leaf images
[47]	2019	Algorithm	Disease	Brinjal	Pseudomonas solanacearum, Cercospora solani na others	Leaf images
[48]	2021	Algorithm	Disease	Paddy, Corn, and Cucumber	Paddy blast and others, Corn Eyespot, and others, Moderate (cucumber) and others	Leaf images
[49]	2021	Algorithm	Disease	Tomato	Early blight and late blight	Leaf images
[50]	2021	Algorithm	Disease	Tomato	Yellow Leaf Curl Virus, Mosaic Virus, Bacterial spot, and others	Leaf images
[51]	2021	Algorithm	Pest	N/A	Whitefly and Thrips	Insect images from a sticky trap
[52]	2021	Algorithm	Disease	Tea	Algae leaf spot, gray rust, white spot, and others	Leaf images
[54]	2021	Algorithm	Pest	Mustard and fava bean	N/A	Insects' images
[53]	2021	Algorithm	Disease	N/A	Multiple diseases	Leaf images
[55]	2021	Algorithm	Disease	Citrus	N/A	Leaf images

[56]	2021	Algorithm	Disease	Citrus	Anthracnose, Canker, and Citrus scab	Leaf images
[57]	2021	Algorithm	Disease	Citrus	black spot, Canker, scab, greening, and Melanose	Leaf images
[58]	2020	Algorithm	Disease	Coffee	N/A	Leaf images
[59]	2020	Algorithm	Disease	Potato	N/A	Leaf images
[60]	2020	Algorithm	Disease	Cherry, Corn, Grape, Orange, Peach, Pepper, Potato, Raspberry, and Soybean	Several diseases	Leaf images
[61]	2020	Algorithm	Disease	Tomato	N/A	Leaf image
[62]	2020	Algorithm	Disease	N/A	Alternaria alternata	Leaf image
[63]	2020	Algorithm	Disease	Rice	Stackburn, Leaf Scald, Leaf Smut, White Tip, and Bacterial Leaf Streak.	Leaf images
[64]	2020	Algorithm	Disease	Soybean	N/A	Leaf images from UAV
[65]	2020	Algorithm	Disease	Peanut, Paddy, Cotton	Cercospora leaf spot, Bacterial Bligh, and others	Leaf image
[66]	2019	Algorithm	Disease	Sugar beet	Cercospora beticola Sacc.	Leaf images
[67]	2019	Algorithm	Disease	Citrus	Citrus scab, anthracnose, greening, and others.	Leaf images
[68]	2019	Algorithm	Pest	N/A	Several insects	Insects' images
[69]	2019	Mobile Application	Disease	Tomato	Bacterial spot Xanthomonas campestris, mosaic virus, and others	Leaf images

[70]	2019	Algorithm	Disease	Guava	Whitefly, Algal leaf spot, and Rust	Leaf images
[71]	2019	Algorithm	Disease	Cotton	Bacterial Blight, Alternaria Leaf Spot, Gray Mildew, Magnesium Deficiency, and others	Leaf images
[72]	2019	Algorithm	Disease	Grape	Black rot, Esca, and leaf blight	Leaf images
[73]	2018	Algorithm	Disease	Tomato	Septoria leaf spot, Yellow leaf curl	Leaf images
[74]	2018	Algorithm	Pest	Bean, Potato	Mexican Bean Beetle and Colorado Potato Beetle	Fruit images
[75]	2017	Algorithm	Disease and pest	Tomato	Several tomato diseases	Leaf images
[76]	2017	Algorithm	Disease	Rice	Several diseases	Leaf images
[77]	2017	Algorithm	Disease	Bens, Lemon Banana	Several diseases	Leaf images
[78]	2015	Algorithm	Disease	Tomato	Powdery mildew fungus, Oidium neolyopersici	Leaf images
[79]	2021	Algorithm	Disease	Peanut	Rust, scorch, and leaf-spot	Leaf images

Table 4 identifies the dataset, the proposed algorithm, and the accuracy achieved in each one of the algorithms described in each article.

Table 4. Summary of the datasets and algorithms.

Paper	Dataset	Algorithm	Accuracy
[32]	Self-collected dataset	Faster R-CNN	93.2%
[33]	Self-collected dataset (20k images)	EfficientNet-b0 model from CNN	97% or more
[34]	N/A	YOLOv4	97.55%

[35]	Dataset from paddy farmlands situated at UAS, India	VGG-16 and GoogleNet CNN models	92.24% and 91.28%
[36]	Self-collected dataset (12k images)	CNN (VGG 16 + ToLCNDV and ToLCGV)	97.211%
[37]	Kaggle's website - with 10.239 leaf images	k-NN approach (LBP, HOG, TEM, GWT, and GLCM)	81.55% - 89.7%
[38]	PlantDoc (2,567 images)	Generative Adversarial Networks (GAN modules)	51.7% (<i>Precision</i>), 48.1% (<i>Recall</i>), and 50.3% (<i>mAP</i>)
[39]	Self-collected Dataset	Fitness-distance balance deep neural network (FDB-DNN)	2.8140%, 1.507%, 5.125%, and 0.402% higher than the existing state-of-art
[40]	Self-collected dataset	Faster R-CNN, Mask R-CNN	31% to 96%
[41]	Self-collected dataset (3.900 images)	MIB Classifier	98%
[42]	From Kaggle's website	Five-layered CNN model	99.99%
[43]	PlantVillage	Own CNN model with 18 layers and 28 target classes	99.12%
[44]	Self-collected dataset	Cascade R-CNN model with Res2Net	81.0% (mAP)
[45]	Self-collected dataset	CNN models (VGG16 & VGG19)	VGG16 (98.52%) and VGG19 (98.08%)
[46]	Self-collected dataset (50k images)	CNN + k-NN Classifier + SVM Classifier + Random Forest Classifier	99.2%
[47]	Self-collected dataset (brinjal leaf images)	ANN Classifiers	91% (with FFNN) and 98.1% (with CFNN)
[48]	PlantVillage	CNN models (MobileNet-V1, MobileNet-V2, NASNetMobile, EfficientNet-B0, and DenseNet-121)	98.44% and 99.46%
[49]	Self-collected dataset	Transfer Learning model	99.86%

			99.89% (more significant result with EfficientNet-b4)
[50]	PlantVillage	EfficientNet families	
[51]	Self-collected dataset	Faster R-CNN	0.944 and 0.952
[52]	CIFAR-10 dataset	CNN model with 1 input layer	94.45%
[54]	Self-collected dataset	ORB algorithm	91.89%
[53]	PlantPathology	VGG16 DL model	85%
[55]	N/A	CNN model	More than 90%
[56]	Self-collected dataset	Models: MobileNetv2 and DenseNet201	95.7%
[57]	PlantVillage	Multi-Layer CNN	94.55%
[58]	PlantVillage and Coffee Leaf dataset	ResNet18, ResNet34 e ResNet50 (models from CNN)	99% and 81%
[59]	PlantVillage	ML classifiers (Random Forest, k-NN, Decision Trees, SVM, and others)	97% (higher precision from RF)
[60]	PlantVillage	FSL method, SVM classifier	94.0% (higher precision)
[61]	Kaggle datasets	SVM, LR, RF	87.6, 67.3 and 70.05
[62]	N/A	K-Means Clustering (for alerts)	95.1613%
[63]	ImageNet and PlantVillage	VGGNet	91.82% and 92.0%
[64]	Images taken from UAV (drone)	DL models (ResNet-50, Xception, VGG-19 and Inception-v3)	99.04%
[65]	PlantVillage	Enhanced Fusion Fractal Texture Analysis (EFFTA)	97.69%
[66]	Self-collected dataset	Faster R-CNN models	95.48%
[67]	Kaggle datasets	SVM + ANN	SVM (93.12%) and ANN (88.96)
[68]	NBAIR dataset, Xie1 and Xie2 dataset	Self-model proposed is based on CNN models with six convolutional layers	96.75, 97.47, and 95.97%
[69]	Plant health (7176 images of tomatoes used)	CNN (MobileNet Model)	90.3%

[70]	Self-collected dataset (BU_Guava_Leaf (BUGL2018))	11 layers-based D-CNN model	98.74%
[71]	Self-collected dataset (images taken by digital camera)	SVM classifier	98.46%
[72]	PlantVillage	SVM, adaboost, and Random Forest tree	93%
[73]	PlantVillage	AlexNet, GoogleNet and LeNet	94% – 95%
[74]	Self-collected dataset	RSC (Random Subspace Classifier) and LIRA (Limited Receptive Area)	88% (CPB with LIRA) and 89% (MBB with RSC)
[75]	Self-collected dataset	VGG16 + SVM	89%
[76]	Heilongjiang Academy of Land Reclamation Sciences, China	CNN model	95.48%
[77]	Self-collected dataset (Images from a digital camera or similar devices)	SVM Classifier	95.71%
[78]	Middlebury dataset	SVM classifier	More than 90%
[79]	Self-collected dataset	CNN (ResNet50 and DenseNet121)	ResNet50 (97.59%) and DenseNet121 (90.50%)

2.5 Discussion

In this section, some details and results of the review are discussed. This discussion will follow the answers to the research questions initially proposed in this review.

Although the search strategy considered works published in the last ten years, after applying the inclusion and exclusion criteria, only studies published since 2017 (last six years) remained to be included in the review. This reveals the interest that this research area has attracted in recent years, but it also indicates that it is still a new area of research.

Most of the studies included in the review (43/48 – 89.6%) describe algorithms and models with a focus on analyzing the performance of these algorithms. Only 10.4%

(5/48) of the analyzed studies present applications (web, mobile, or robotic systems) that end users can use.

There is a significant predominance of ML-based approaches regarding the analyzed algorithms and models.

For each of the research questions identified as the target of this study, described in section 1 below and based on the results of the systematic review (presented earlier), we present answers to each of them below.

The review shows that the crops most focused on studies of automatic detection of diseases and pests (RQ1) are tomatoes and citrus. Tomato crops are studied in 22.5% (9/40) of the analyzed studies. Citrus was analyzed in 15.0% (6/40) of the studies. Rice, grapes, beans, and Corn are in third place, with 7.5% each (3/40). Then come the Apple, Peach, Pepper, Cotton, and Paddy cultures with 5.0% each (2/40), and lastly follow the Cucumber, Banana, Cassava, Peanut, Sunflower, Herbs, Raspberry, Strawberry, Brinjal, Tea, Mustard, Coffee cultures, Soybean, Sugar, Guava, and Lemon with 2.5% (1/40) of the analyzed studies. In this question, only 40 studies were referenced because eight did not have consistent information.

The automatic detection of diseases has attracted more attention than the detection of pests (RQ2). The review shows that 85.7% (40/48) of the studies focus on diseases, and 18.8% (9/48) of studies refer to plagues or pests (one of the studies refers to diseases and pests). The most commonly studied diseases were in the tomato crop (the most studied crop): ToLCNDV & ToLCGV (begomovirus infections), Early blight, and Late blight. For pests or plagues, the most commonly found were Wheat mites, wheat aphids, wheat sawflies, and rice plant hoppers.

All analyzed studies use images of leaves, fruits, plants, or insects as input to their algorithms for detecting diseases and pests. Leaf images were the most common input for detection/identification/classification (RQ3). The analysis shows that 85.4% (41/48) of the studies refer to this input. Next, it is found that 8.3% (4/48) of the studies refer to Insect images and 4.2% (2/48) refer to Fruit, and 4.2% (2/48) to plant images. Additionally, practically all algorithms that use images of leaves use images in which the leaf is the main element of the image. Usually, the image of the leaf occupies practically the entire image area. Only a minority of studies use leaf images from

vehicles such as UAVs. The datasets used for pest detection include leaf images and Insect images from sticky traps.

Generally, the analyzed approaches use image datasets to train and validate the proposed algorithms. In these cases, image datasets that are publicly available can be used, or new datasets can be constructed and used. The analysis of the studies revealed that PlantVillage is the most used dataset (RQ4). It was used in 24.4% (11/45) of the studies. In sequence, 8.9% (4/45) of the studies used the Kaggle dataset. Other datasets were also used, each one in one study: PlantDoc Middlebury dataset, Heilongjiang Academy of Land Reclamation Sciences - China, Plant health, NBAIR dataset, Xie1 and Xie2 dataset, Coffee Leaf dataset, PlantPathology, CIFAR-10 dataset and a dataset from paddy farmlands situated at UAS, India. A significant number of studies, 46.7% (21/45), used self-collected datasets. Three studies did not provide information about the datasets used.

Regarding the algorithm or models most used (RQ5), the review shows that CNN models are the most used for research or studies in this area. About 54.2% (26/48) of the studies refer to CNN models, namely Faster R-CNN, EfficentNet, VGG, GoogleNet, MobileNet, ResNet, AlexNet, LeNet, and DenseNet. Next, 16.7% (8/48) of studies used the SVM classifier, 8.3% (4/48) used the k-NN algorithm, Random Forest is used in 8.3% (4/48), 4.2% (2/48) used ANN classifiers. Some studies proposed approaches with the following algorithms: YOLOv4, ORB algorithm, RSC (Random Subspace Classifier), LIRA (Limited Receptive Area), K-Means Clustering, Decision Trees, MIB Classifier, GAN modules, and Enhanced Fusion Fractal Texture Analysis.

The level of accuracy achieved in detecting, identifying, and classifying diseases or pests in agricultural crops (RQ6) depends on several factors, namely the crop and the diseases and pests considered. Higher levels of accuracy were achieved in the three crops mentioned in the studies. In the first case, tomato crops, the highest precisions found vary from 90.3% to 99.89%. Next, in the second case, the citrus crop, the accuracies vary from 88.96% to 98%. Finally, the accuracy in the potato crop varies from 89% to 97%. It is also essential to keep in mind that these values depend on the disease or pest being studied.

This review allowed us to identify some trends that have been observed either because they are used in a significant number of studies or because there has been a growing interest in their use over the years (RQ7). CNN-based algorithms tend to be

the most used to achieve the objectives in this area (see Figure 1 left). ML-based approaches are becoming popular in developing solutions for plant diseases and pest detection. Approximately 93.8% (45/48) of the studies in this review proposed ML-related approaches. Also, the number of articles related to ML-based approaches tends to increase yearly, indicating its popularity and may grow even more. A deeper analysis shows that in the last decade, the most used algorithms began to be cited in studies from 2017 onwards. An increase in CNN over the other algorithms was subsequently noticed (see Figure 1, right).

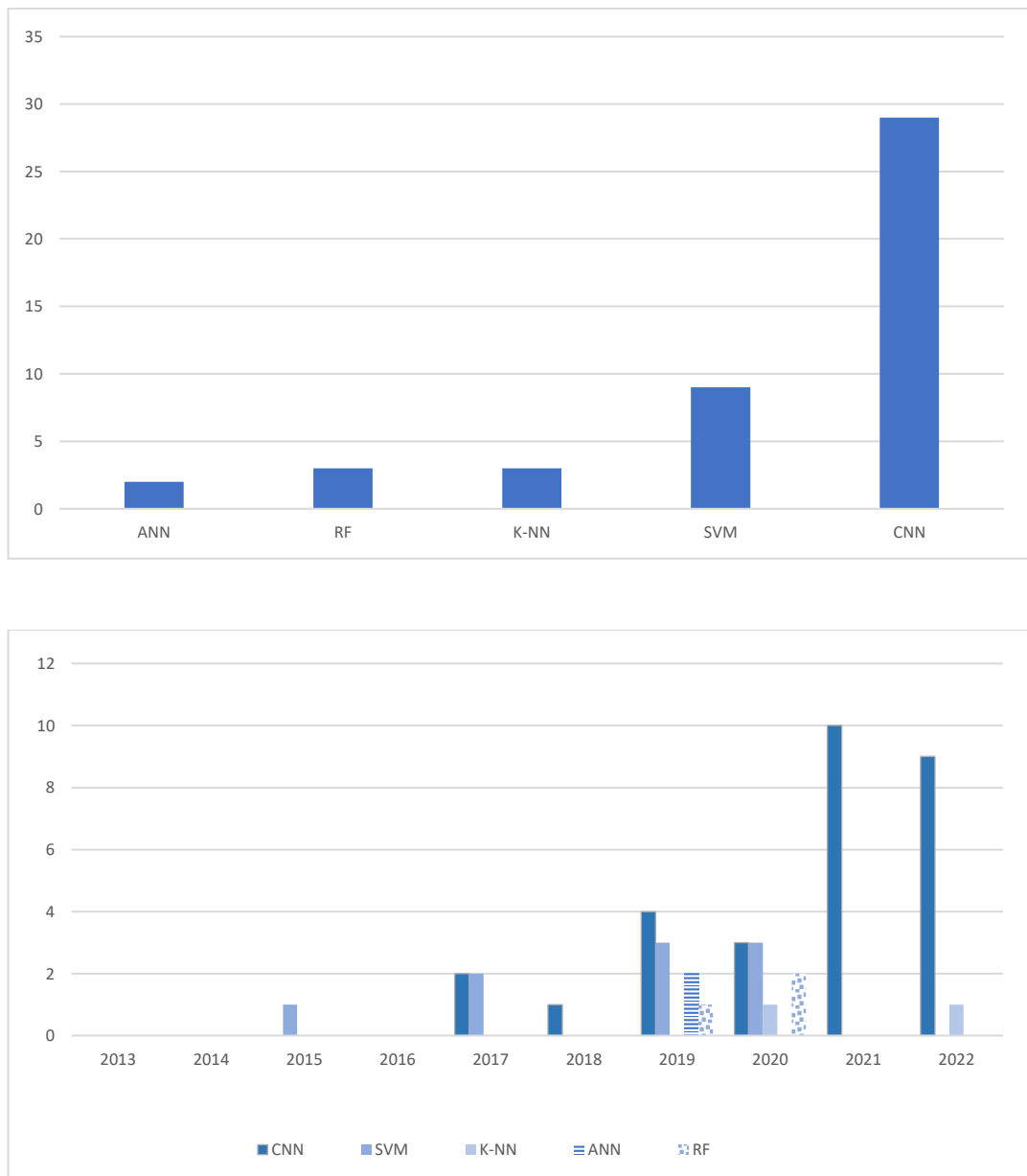


Figure 2. *Top)* Most used models. *Bottom)* Most used models over the years.

Accuracy highly depends on the crops, diseases, and pests considered, the algorithms used, and the datasets used. However, the results show that it is possible to achieve very high levels of accuracy in some diseases and pests in several crops. This indicates that these approaches can be used in applications that can be used in natural environments. Most of the datasets represent images of leaves taken with cameras, where the leaves are the central object of the photo, although the backgrounds can vary. There are still very few cases in which the images are obtained using cameras in land or air vehicles, which would represent a step forward in obtaining solutions that are more adequate to the reality of agricultural crops.

2.6 Strengths and Limitations

This review followed the PRISMA methodology. It provides a systematic review of the existing works that approach automatic detection, identification, and classification of diseases and pests in agricultural crops. This review is critical because it presents an overview of the most studied crops and a characterization of the algorithms and models used, their inputs, the datasets most used to train and validate them, and the accuracy achieved. In addition, it also presents some trends that have been observed. It represents an essential basis for academics and researchers to understand this study area and develop new algorithms and applications.

However, it also has some limitations. The literature search was carried out using two databases (Scopus and Web of Science). These databases cover several domains and span many individual databases. However, other databases, such as IEEE Xplore, ACM Digital Library, PubMed, ScienceDirect, or BMC, could have led to more articles being included in the review. The search strategy may have influenced the number of articles considered in the study. For example, the search string used, the option to search only for articles written in English or only for articles published in the last ten years, may also have influenced the number of relevant articles considered. Although these limitations may have affected the number of articles obtained and considered in the review, we believe these constraints did not significantly affect the discussion and conclusions.

2.7 Related Applications

It is crucial to recognize the practical uses of these technologies in current mobile applications, in addition to thoroughly reviewing algorithms and processes. These applications use machine learning and computer vision to detect diseases in agricultural crops by analyzing their leaves. For this study, a number of applications were chosen that closely reflect the concept of the proposed prototype. The purpose was to gain a deeper understanding of their functions and contributions. Following it is a detailed description of the applications and their respective functions.

Plantix [80] - This application offers a robust solution for image-based plant disease identification. Using an artificial intelligence system, Plantix analyzes the photos sent and provides accurate diagnoses, as well as treatment recommendations. Additionally, the app includes an active community where users can interact with experienced agronomists for discussions and additional questions.

PictureThis [81] - With the ability to identify more than 17,000 plant species with 98% accuracy, this application goes beyond simple identification by offering automatic disease diagnoses. This app also includes plant care reminders, expert consultations and toxic plant alerts, acting as a complete crop care assistant.

Leaf Doctor [82] - This application uses an approach based on analyzing leaf color to identify plant health. Users upload a photo of the leaf, and the app assesses the degree of damage based on predefined colors, providing a percentage of diseased tissue that helps determine the stage of the disease and the effectiveness of possible treatments.

Agronomist.ai [83] - This application uses deep learning models to identify diseases in crops, standing out for its ability to process images of leaves and diagnose a variety of diseases. Agronomist.ai offers support through educational videos and instant solutions to mitigate losses.

These applications exemplify the real implementation and efficacy of the mentioned approaches, offering useful insights for the development of the pretended system/prototype.

2.8 Final Remarks

The review presented in this chapter aimed to find studies on automating processes in detecting, identifying and classifying diseases and pests in agricultural crops. It followed the PRISMA methodology. The literature search was conducted in two abstract and citation databases (Scopus and Web of Science). The initial search returned 278 studies, and after removing duplicates and applying the inclusion and exclusion criteria, 48 articles were included in the review. All analyzed studies propose the research structure related to the detection, identification, and automatic classification of diseases or pests in agricultural crops. However, the objective of this study, in addition, to presenting the review, identifying the most studied crops, characterizing the proposed algorithms, results achieved, and the most used datasets, the aim is also to present a document to support researchers who intend to develop work in this area, characterizing this area of study and identifying some of the most noted trends. In this sense, it represents a step forward in relation to other related works, thus representing a significant contribution to this study area.

The results indicated that most of the studies were focused on algorithms or systems that allow the presentation of results using the various Deep and ML techniques and that 95% of the studies focus on demonstrating the ability of specific algorithms and models in solving problems related to the automatic detection of diseases or pests. In all cases, it was necessary to use a dataset. Analysis showed that PlantVillage dataset was the most used. Models and classifiers such as CNN, SVM, k-NN, ANN, Random Forest, and others were used to train the datasets, classify the diseases and pests, and achieve better accuracy for each algorithm. The accuracy achieved depends on the diseases and pests in the agricultural crops.

This review also made it possible to identify some gaps in information in some contents, which caused difficulties in the research. More specifically, in some cases, they did not provide enough information about the dataset they used. In some cases, researchers were not careful to mention the crop or even the diseases and pests analyzed in the study, thus making it difficult to collect the exact information. The application designed for this dissertation not only includes identical functionality as indicated, but also introduces other features. Furthermore, it enables the generation of comprehensive reports that encompass the identification of the impacted crop,

geolocation data, the date of report production, and privacy settings to either share or maintain the confidentiality of the information. This supplementary capability is specifically developed to aid in strategic decision-making and efficient crop management, offering a valuable instrument for farmers and farm managers.

The objective of this study is to use machine learning methods to create a prototype system capable of autonomously detecting diseases in agricultural crops using leaf photographs. The primary objective of this system is to enhance the precision of disease detection and facilitate timely intervention, a critical factor in mitigating crop losses and promoting sustainable agricultural methods. An administration panel is included to assist users in reporting, saving, and evaluating data concerning recognized diseases, pests, and anomalies. This, in turn, helps in making educated decisions and effectively managing crops. In addition, the system utilizes geolocation capabilities to capture the precise latitude and longitude of every identified disease, pest, or anomalies report. This feature enables the mapping of the locations where each report was conducted, allowing users to visually depict the geolocation distribution. The system aims to enhance the capacity to monitor the progression of issues over time and perform targeted interventions in areas with the greatest need by utilizing geolocated data. The geolocation feature of the suggested solution is a notable benefit, enhancing the level of knowledge and strategic planning in crop management.

3. Requirements and Modeling

This chapter is organized into multiple sections to facilitate understanding of the system requirements and the modeling process of this system.

Section 3.1 explores the two main classifications of requirements: functional and non-functional. It emphasizes their significance in constructing software systems that efficiently satisfy user requirements and fulfill stakeholder expectations. Functional requirements specify the precise activities, behaviors, and functionalities that the system must have. They help developers create the necessary elements for the system to carry out tasks like data processing, user authentication, and reporting. Non-functional requirements, in contrast, provide the operational criteria for the system, highlighting its performance in carrying out functions rather than the precise tasks it carries out. These encompass essential factors such as the dependability of the system, measurements of performance, adherence to security standards, and considerations of usability.

Section 3.2 employs use case diagrams as a visual tool to illustrate and understand these system requirements. These diagrams visually depict the likely interactions between users and the system. They help to understand the requirements and enhance communication among all stakeholders involved in the project.

Section 3.3 offers comprehensive explanations of each use case, outlining the activities and anticipated results for various system actors, including Admins, Team Leaders, and Team Members.

Section 3.4 presents the storyboards that were created throughout the design phase. These illustrations function as a plan for the system's user interface, offering detailed information about every aspect, such as navigation menus, button placement, page layouts, and user interaction sequences.

Section 3.5 discusses the Entity-Relationship (ER) Data Model. This section delves into the ER Data Model, an essential tool for the modeling and development of databases. The ER model is used to represent the entities in the issue domain and their relationships, providing a comprehensive view of the data structure and facilitating clear communication and understanding of the system among stakeholders.

This chapter gives understanding of the value of use case diagrams and functional and non-functional requirements for creating software that generally satisfies project goals and end user expectations.

3.1 Requirements

This section examines the two primary categories of system requirements: functional and non-functional. It emphasizes their significance in constructing software systems that efficiently fulfill the demands of users and meet the expectations of stakeholders.

Functional requirements delineate the precise activities, behaviors, and functionalities that the system is obligated to possess. They play a vital role in providing guidance to developers in designing the necessary functionality for the system to carry out specific activities, such as data processing, user authentication, and reporting. Their mission is to ensure that the software accomplishes its fundamental objective.

In contrast, non-functional requirements define the system's operational standards, emphasizing how it executes its functions rather than the specific tasks it carries out. These encompass crucial factors such as the dependability of the system, measurements of performance, adherence to security standards, and considerations of usability. While a system can operate without fully satisfying all non-functional needs, neglecting certain factors might result in user unhappiness and subpar system performance [84].

Use cases diagrams are utilized as a graphical instrument to depict and comprehend these requirements. They provide a depiction of users' probable interactions with the system, aiding in the comprehension of requirements and improving communication among all stakeholders engaged in the project [85].

This chapter provides a comprehensive explanation of the significance of each category of requirement in the software development process. It covers several methodologies and optimal approaches for identifying, documenting, and managing these requirements. Our goal is to offer a comprehensive perspective on the importance of functional and non-functional requirements, as well as use case

diagrams, in developing efficient and dependable software systems that meet end-user expectations and project objectives.

Below, was provided a detailed analysis on the functional and non-functional requirements of this system, drawing on additional resources from [84] which provide comprehensive discussions on these topics.

It is essential to mention that the system is centered around three main actors, each fulfilling a vital function in its functioning:

- The Admin is responsible for supervising the entire system, including managing user accounts, configurations, and ensuring the system operates smoothly. They possess the utmost amount of authority and possess the capability to provide thorough reports, retrieve all data, and formulate strategic judgments.
- Team Leader oversees their individual teams, delegating duties and reviewing the progress. They may obtain data and reports that are relevant to their team's activities, which helps them make well-informed decisions and coordinate effectively.
- The individual in this role is responsible for carrying out actions on the field, such as taking pictures of leaves and documenting data. The Mobile Application is used to gather precise and timely data, essential for the system's analytical procedures.

The system establishes definitions for these roles and their respective functions, resulting in a well-organized workflow that improves the efficiency and effectiveness of disease control and crop health monitoring.

3.1.1 Functional Requirements

This subsection provides an explanation of the primary functions that the system must do. These functions are derived from analyzing user requirements and the objectives specified in the project scope. The requirements will encompass multiple facets of the system, encompassing data intake, processing, output, and user interactions. The goal is to establish precise standards that will guarantee the system meets users' expectations and offers the essential capabilities for efficiently detecting and managing diseases in agricultural settings. In order to illustrate the functional requirements of the system, we have opted to display them in relation to everyone who plays a role in the system, specifically, the Admin, Team Leader, and Team Member. Here is a compilation of the system's functional requirements:

Admin Management (Web)

FR-1: In this solution, to access the system, the functional authentication requirement requires that the Admin user must log in, and out (when leaving the system). Before authenticating, you can register on the platform.

FR-2: The system must allow the administrator to search, register, edit and delete other administrators.

FR-3: The system must allow the administrator to manage team leaders, with features for searching, registering, editing and deleting.

FR-3: The system must allow the administrator to manage items and fields, including features for searching, registering, editing and deleting.

Team Leader Management (Web)

FR-4: The system must allow the team leader to create, edit, search, assign members and delete teams.

FR-3: The system must allow the team leader to manage team members, with functionality for creating, editing, searching and deleting.

FR-4: The system must allow the team leader to manage reports, with functionality for creating, editing, searching, viewing details and deleting.

Team Member Functionalities (Web and Mobile)

FR-5: The system should allow the team members to log in via user credentials.

FR-6: The system must allow the team members to report diseases, pests, anomalies and suggest improvements in agricultural crops.

FR-7: The system must provide automatic disease recognition through machine learning.

FR-8: The system should allow the team members to view a list of reports made and see details of these reports.

FR-9: The system must present a map with geolocation of the reports made.

3.1.2 Non-functional Requirements

This subsection outlines the primary non-functional requirements that the system must fulfill. Generally, the concept is to present each of the non-functional requirements in the system as individual pieces. These items were mentioned: usability, reliability, scalability, location, compatibility, performance, and security. These criteria will provide direction for the design and implementation stages, ensuring all crucial elements of system performance and quality are considered and resolved.

Here is a compilation of non-functional system requirements:

Usability

NFR1.1: The system must be intuitive and easy to use, requiring minimal training for users.

NFR1.2: The system must offer a responsive interface that adapts to different devices and screen sizes.

Performance

NFR2.1: The system must perform automatic disease recognition in a response time of less than 5 seconds.

NFR2.2: The system must be capable of supporting up to 1000 simultaneous users without significant performance degradation.

Security:

NFR3.1: The system must implement user authentication and ensure the security of login data.

NFR3.2: The system must encrypt the transmission of sensitive data.

Scalability

NFR4.1: The system must be designed to facilitate the addition of new features and functionality without major modifications to the existing architecture.

Reliability

NFR5.1: The system must be available 99.5% of the time, with the exception of scheduled maintenance.

Compatibility

NFR6.1: The system must be compatible with the main web browsers and mobile operating system versions.

Location

NFR7.1: The system must support precise geolocation, with a margin of error of less than 10 meters.

3.2 Use Cases Diagrams

The use cases diagrams section provides a visual representation of the functional needs of a system. These diagrams, which falls under the Unified Modeling Language (UML) category, are specifically created to illustrate the interactions between various actors (users or external systems) and the system in order to accomplish specific objectives. Actors are entities that engage with the system, whereas use cases are the specific functions or processes that the system can execute in response to the actor's requests. The diagram is valuable because it provides a comprehensive overview of the system design, making it easier to comprehend the system's features and how users interact with it [86].

Use cases diagrams are created in the first phases of the software development process to capture the system requirements. They serve to verify the system design, assist in implementation, and provide test cases. Because of their simplicity, they function as a powerful means of communication among stakeholders, enabling them to more easily comprehend and visualize the proposed system design.

To gain a comprehensive understanding of use case diagrams and their efficient structuring, including the identification of actors and use cases, these resources and examples can be consulted and are made available by [86]. These resources provide valuable insights into the practical application of these diagrams in systems development.

Therefore, the use case diagrams considered throughout the system's development for this project are now presented, aligning with the previously mentioned functional requirements. It analyzes the diagram that shows the hierarchical connection between the Administrator, the Team Leader, and the Team Member. The graphics depict the activities conducted by the Administrator and the Team Leader on the website, while the actions of the Team Member are presented in the Mobile App, which is their primary area of focus.

Figure 3 presents the use case diagram that pertains to the Admin, who is one of the actors in the system. This diagram illustrates the several use cases allocated to the Admin, which are the functionalities accessible in the system that this actor will be able to utilize.

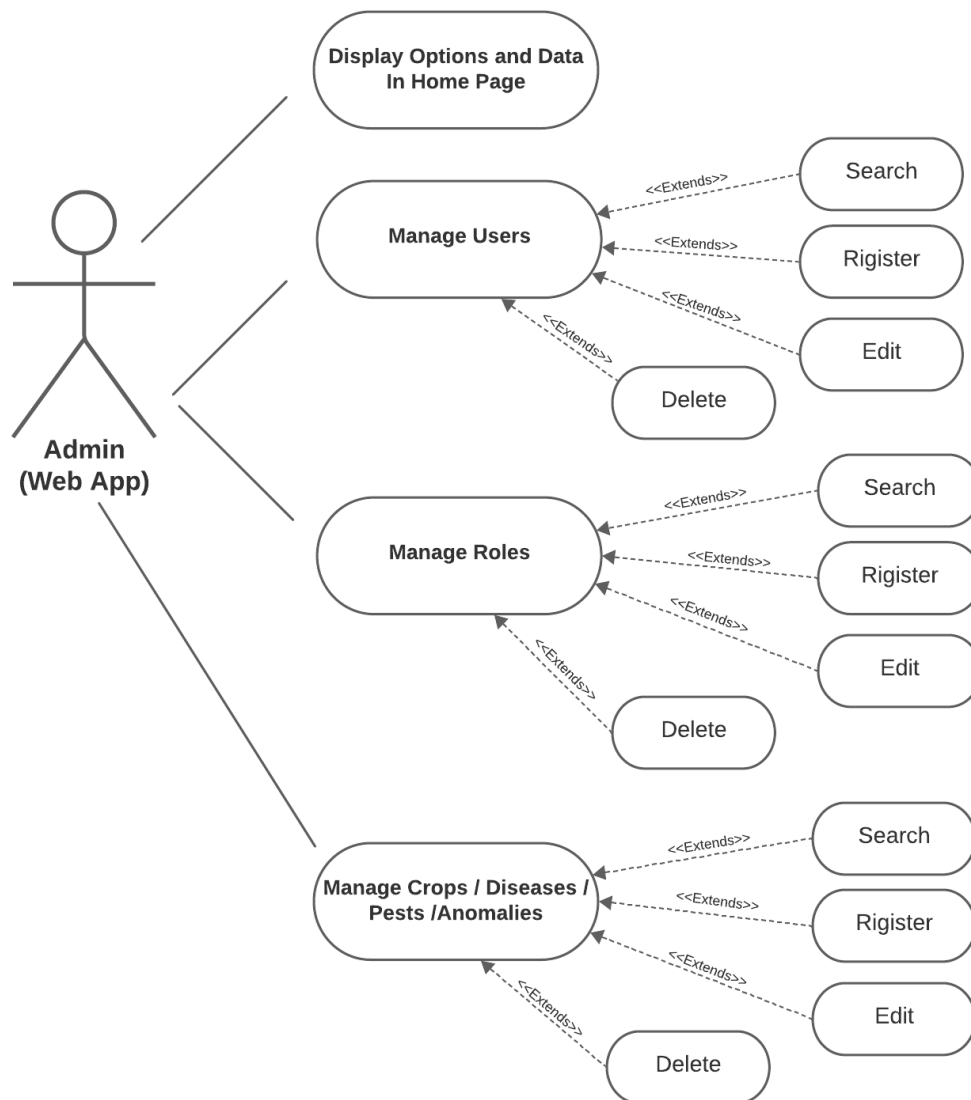


Figure 3. Admin use cases diagram.

Figure 4 presents the use case diagram that pertains to the Team Leader, who is one of the actors within the system. This diagram depicts the various use cases assigned to the Team Leader, which represents the system's accessible functions that this actor can utilize.

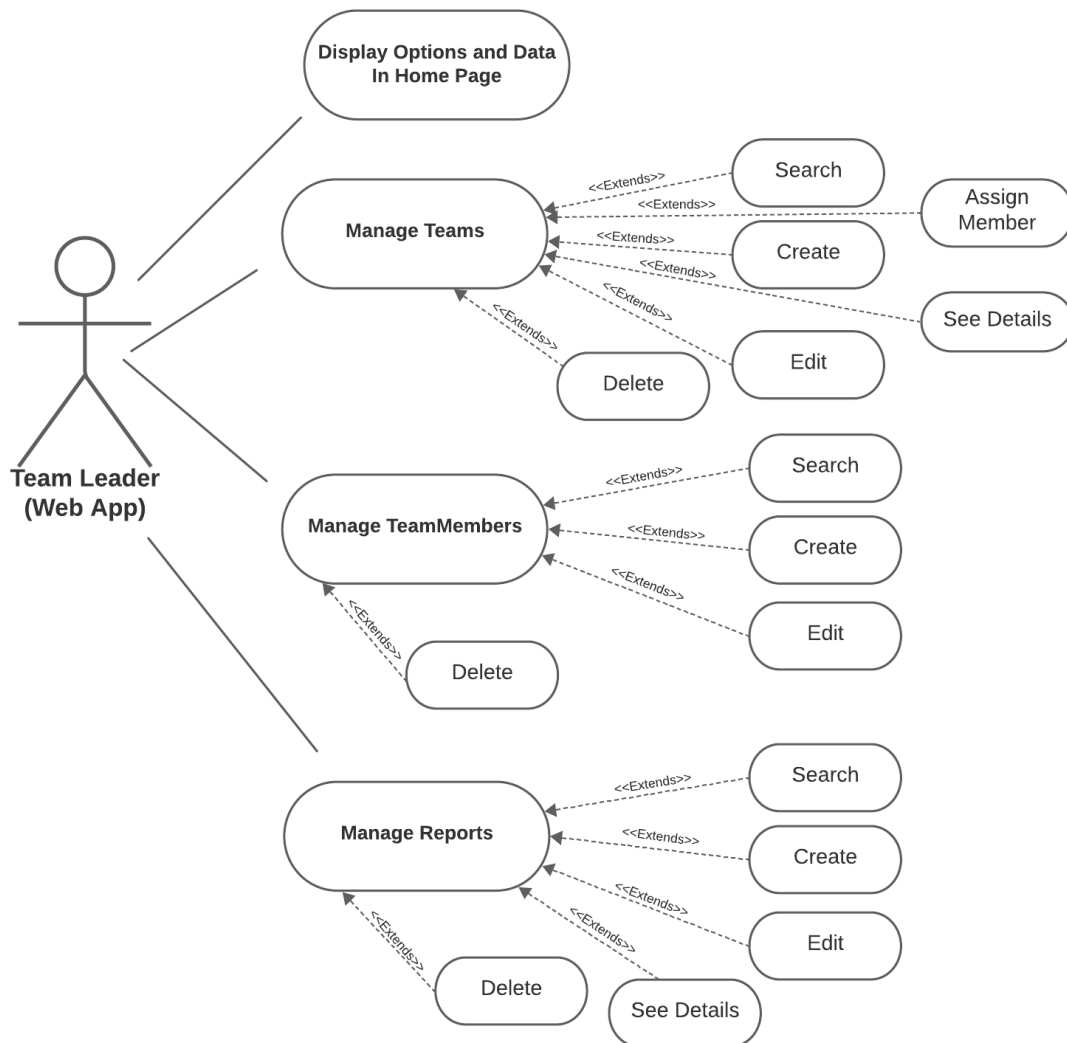


Figure 4. Team Leader use cases diagram.

Figure 5 presents the use case diagram illustrating the involvement of the Team Member, who is both an actor in the system and a field operator. This diagram shows the different use cases allocated to the Team Member, which denotes the system functions this actor can use.

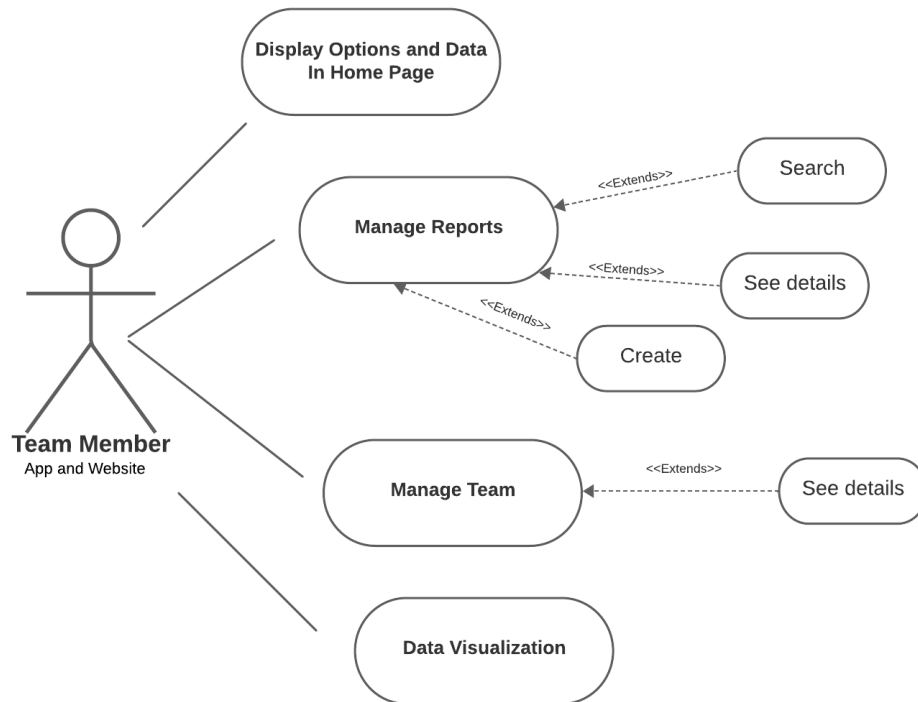


Figure 5. Team Member use cases diagram.

3.3 Use Cases Description

This part outlines the use cases identified in the provided use case diagram, adhering to the structure established earlier in this study. This part is further subdivided into subsections, wherein each use case is comprehensively analyzed.

It is crucial to acknowledge that the use cases mentioned in this context have been simplified to prioritize clarity and emphasize the fundamental features of the system. This approach enables the provision of a scenario illustrating the interactions between various user functions and the system. The use cases were generated utilizing specific components deemed crucial for the comprehensive depiction of each situation, encompassing the primary individuals engaged, the sequence of events, and the anticipated outcomes.

The use cases were described using a simplified manner, with certain factors included for each use case description. These parameters include Primary Actor, Preconditions, Guarantees, and Success Scenario. This guarantees that the use cases are readily comprehensible and directly applicable to the stages of development and execution. Furthermore, these reduced scenarios serve as a strong basis for subsequent improvement and growth as the system progresses and additional needs are recognized.

The primary objective is to execute the correct strategy that guarantees each scenario has the essential interactions and functions. This method streamlines the development of a system that is easy for users to navigate and is specifically designed to identify and locate diseases, pests, and anomalies in agricultural crops.

3.3.1 Admin use cases

This subsection provides a detailed explanation of the use cases depicted in the diagram involving the Admin actor. A simpler version was made to describe the use cases, enhancing the visibility of comprehension based on the developed prototype.

Here is a list of the described use cases:

Display Options and Data in Home Page

The purpose of the "Display Options and Data in Home Page" use case is to provide the administrator with a concise overview of the system's state by utilizing the primary dashboard. The Administrator can access settings and menus and view the count of users, teams, and reports in the system. This information is displayed in a user-friendly format, using intuitive cards and interactive images.

- Primary Actor: Admin
- Preconditions: Admin user must be authenticated with the appropriate privileges.
- Guarantee: Immediate and insightful overview of system statistics.
- Success Scenario:
 - The admin logs into the system and is directed to the main dashboard;
 - The system displays cards and graphs indicating the total number of users, teams, and reports;
 - The cards that appear with information are also the options available for the Admin to see Disease Reports, Pest Reports and Anomaly Reports
 - The admin reviews these metrics for a comprehensive understanding of system activity;
 - The system ensures the data is current and allows for interactive engagement with the graphs for more detailed insights.

Manage Users

This use case enables the Admin actor to carry out essential administrative tasks, including searching for, registering, modifying, and deleting administrator profiles. These tasks are vital for preserving the team's structural integrity and managing access permissions in the system.

- Primary Actor: Admin
- Preconditions: Admin user must be authenticated with the appropriate privileges.
- Guarantee: Effective administration of admin roles and permissions.
- Success Scenario:
 - The admin navigates to the "Manage Users" functionality;
 - The system displays interface options for "Search", "Register", "Edit", and "Delete";
 - The admin selects and executes the desired action;
 - Clicking should trigger a pop-up for all CRUD actions. When using the Create function, the form will display the required fields for registering a new user. This includes a multiple-choice option that allows the user to assign a Role to the newly registered user. Upon completion, a success notification is presented, allowing you to either close the pop-up or proceed on the current screen to register additional users.
 - For Update, the idea is the same, launch the pop-up with the data in the form and you can change any information as needed. And when you click on "Save Changes" you will receive a success message and then close the pop-up.
 - For Delete, upon clicking the operation, a pop-up should appear prompting you to confirm the deletion of the record. If you click on Delete, the system will send you to the view where the updated table of users is placed, along with a success message.
 - After operations, the system updates to reflect the changes.

Manage Roles

The "Manage Roles" feature allows the Administrator to supervise user roles, making it easier to search, register, change, and delete roles. This ensures that the system's leadership structure is up-to-date and operational.

- Primary Actor: Admin
- Preconditions: Admin user must be authenticated with the appropriate privileges.
- Guarantee: Accurate maintenance of roles records.
- Success Scenario:
 - The admin accesses the management section for roles;
 - The system offers options for "Search", "Register", "Edit" and "Delete", the common CRUD operations, and based on the same idea as pop-ups, the behavior for managing Roles works in the same way.
 - The system verifies and applies the changes.

Manage Crops, Diseases, Pests and Anomalies

In the "Manage Crops, Diseases, Pests and Anomalies" section, the Admin can alter or create new items such as Crops, Diseases, Pests, and Anomalies. This use case is crucial for ensuring that the application's content remains pertinent and current by including the most recent agricultural insights and data.

- Primary Actor: Admin
- Preconditions: Admin user must be authenticated with the appropriate privileges.
- Guarantee: The system's content reflects current and accurate agricultural data and recommendations.
- Success Scenario:
 - The Admin enters each area to manage Crops, Diseases, Pests and Anomalies (each with its own menu).
 - The system presents options for "Search", "Register", "Edit" and "Delete", in the same order as in the previous cases. The idea is to open the pop-up to carry out the operations and when submitting the form with the desired components you will then receive a success

message. In Delete, open the pop-up for confirmation and only then can you click on delete to confirm the operation;

- For diseases and pests, the behavior will be similar. The form should have a combobox that will have the list of Crops already inserted into the system. In each Disease or Pest record, it must be mandatory to select which crop they belong to via the combobox.
- The system confirms the completion of the updates.

These use cases collectively form a comprehensive management suite that enables the Admin to maintain the system efficiently, ensuring that both the system's data and the user hierarchy are well-organized and reflective of current operations and knowledge in the field.

3.3.2 Team Leader use cases

This subsection provides a detailed explanation of the use cases depicted in the diagram involving the Team Leader actor.

Here is a list of the described use cases:

Display Options and Data in Home Page

The "Display options and data on home page" use case is designed for the Team Leader to get a snapshot of the system status through the main dashboard. It allows you to view access options and menus, and you can see the total public reports and reports created by you and your team members presented through intuitive cards and interactive graphics, you can also see an interactive calendar with the reports on each date carried out.

- Primary Actor: Team Leader
- Preconditions: Team Leader must be authenticated with the appropriate privileges.
- Guarantee: Immediate and insightful overview of system statistics.
- Success Scenario:

- Team Leader logs into the system and is directed to the main dashboard;
- The system displays cards with the desired options such as Disease Reports, Pest Reports, Anomaly Reports and Suggestions, and in them the totals of each report already carried out can be seen, as mentioned above;
- Team Leader reviews these metrics for a comprehensive understanding of system activity;
- The system ensures that data is up to date and allows interactive engagement via calendar to obtain more detailed insights regarding the dates on which each report was created.

Manage Teams

In this use case, the Team Leader is equipped to orchestrate the configuration of teams within the web application. The Team Leader can perform various tasks to ensure that teams are structured effectively and aligned with project goals.

- Primary Actor: Team Leader
- Preconditions: Team Leader must be authenticated with the appropriate privileges.
- Guarantee: Consistent and accurate management of team configurations.
- Success Scenario:
 - The Team Leader selects the "Manage Teams" option within the web app;
 - The system presents the Team Leader with options to "Search", "Create", "Edit", "Delete", "Assign Member", and "See Details";
 - Similarly, CRUD operations for this use case will utilize pop-ups in a consistent manner. When making updates to a Team, the Team Leader will be able to link a member to the team by using a multiple choice field that allows for the addition of one or more members.
 - The system confirms and reflects the changes to maintain an up-to-date team structure.

Manage Team Members

The Team Leader manages the profiles of each team member. This involves generating new records for team members, modifying their data, and ensuring that only present members possess authorization to the system and its assets.

- Primary Actor: Team Leader
- Preconditions: Team Leader must be authenticated with the appropriate privileges.
- Guarantee: Proper management of team member records.
- Success Scenario:
 - The Team Leader accesses the "Manage Team Members" functionality;
 - The system offers "Search", "Create", "Edit", and "Delete" functionalities for managing team member data, following a consistent pattern like pop-ups and success notifications. Furthermore, the team leader can use a feature inside the interface to invite team members. By selecting the "invite by link" button, a pop-up window will appear, presenting a form where the team leader can input the email address of the individual they wish to join the team;
 - The Team Leader completes the necessary action to manage the team members;
 - The system updates to ensure accurate team member records are maintained.

Manage Reports

The Team Leader possesses the jurisdiction to oversee illness, pest, and anomaly reports documented by both him and his team. This position entails the creation and maintenance of reports and a thorough examination of the data within the reports, including geolocations shown on a map and statistical graphs.

- Primary Actor: Team Leader
- Preconditions: Team Leader must be authenticated with the appropriate privileges.

- Guarantee: Comprehensive and actionable reporting for decision-making.
- Success Scenario:
 - The Team Leader accesses the system and has access to the report management dashboard. The user can select from different categories, including "Disease Reports", "Pests Reports", and "Anomalies Reports", by using cards;
 - By selecting the desired option, you will gain access to your management interface, where the team leader can perform actions such as creating, editing, deleting, viewing details, and visualizing data through graphs and geolocation;
 - By selecting the disease or pest reporting option, the Team Leader can access comprehensive reports that contain interactive maps indicating the exact location of public reports and reports from their team. They can also analyze the data using graphs that emphasize the most significant crops and diseases mentioned in the reports;
 - In this use case, CRUD operations are implemented using pop-up windows that display forms for data entry and provide success messages upon completion of the activity. When generating the report, it is necessary to include an interactive feature that enables the selection of a specific crop for data collection. This feature should automatically display only the diseases associated with the selected crop. This can be achieved using a combobox, as the crops and diseases or pests would already be registered in the system at this stage. Also, the system should automatically capture and include the report's geolocation and creation date. The pattern is identical in both diseases and pests;
 - Typically, while creating a disease report or pest report, you should be able to include a photo, choose the specific crop and disease, and indicate if it will be private or public using a Boolean option.

- The system guarantees the accessibility and accuracy of the report data for the Team Leader's examination and further steps.

3.3.3 Team Member Use Cases

This subsection provides a detailed explanation of the use cases depicted in the diagram involving the Team Member actor.

Here is a list of the described use cases:

Display Options and Data in Home Page

The purpose of the "Display options and data on home page" use case is to provide the Team Member with a concise overview of the system's state by utilizing the main dashboard. The feature enables users to access and navigate through many options and menus. It provides a comprehensive overview of the total number of public reports and reports made by the user. This information is presented in a user-friendly format, using intuitive cards and interactive visuals. Additionally, users can view an interactive calendar that displays the reports conducted on specific dates.

- Primary Actor: Team Leader
- Preconditions: Team Leader must be authenticated with the appropriate privileges.
- Guarantee: Immediate and insightful overview of system statistics.
- Success Scenario:
 - A team member accesses the system and is automatically sent to the main dashboard;
 - The system presents cards with selectable options such as Disease Reports, Pest Reports, Anomaly Reports, and Suggestions. These cards reflect the overall number of each completed report, as previously mentioned;
 - Team members analyze these indicators to have a thorough picture of system activities;

- The solution guarantees the accuracy of the data and enables users to interactively connect with a calendar to gain more comprehensive insights into the specific dates when each report was generated.

Manage Reports (Website)

This use case outlines the team member's ability to engage with reports on the website, with a specific emphasis on search functionality, report generation, and access to comprehensive information.

- Primary Actor: Team Member
- Preconditions: Team Member must be authenticated with the appropriate privileges.
- Guarantee: Comprehensive and actionable reporting for decision-making.
- Success Scenario:
 - A team member logs into the system and gains access to the report management dashboard. He can select the preferred option from the available choices of "Disease Reports", "Pests Reports", and "Anomalies Reports" using cards;
 - By selecting the desired choice, you will gain access to your management interface, where team members may produce and see detailed information and analyze data using graphs and geolocation;
 - By selecting the disease or pest reporting option, the Team Member can access comprehensive reports that contain interactive maps indicating the exact location of public reports and reports from their own team. They can also analyze the data using graphs that emphasize the main crops and diseases mentioned in the reports;
 - In this use case, the Create operation is implemented in a similar manner. It involves a pop-up window with a form, and upon successful completion of the action, a success message is displayed. When generating the report, it is necessary to include

an interactive feature that enables the selection of a specific crop for data collection. This feature should automatically display only the diseases associated with the selected crop. This can be achieved using a combobox, given that the crops and diseases or pests will already be registered in the system. Also, the system should automatically capture and include the report's geolocation and creation date. The pattern is identical in both illnesses and pests;

- Typically, while generating a disease report or pest report, you should have the ability to include a photo, choose the specific crop and illness, and indicate whether the report will be private or public using a Boolean option;
- The system guarantees the accessibility and accuracy of the report data for the Team Leader's examination and response.

Manage Reports (Mobile App)

In this case involving mobile devices, the team member uses the program to generate reports with added features, such as uploading photographs and automatically incorporating GPS data, along with the date and time of report creation.

- Primary Actor: Team Member
- Preconditions: Team Member is authenticated and has mobile app access with necessary permissions.
- Guarantee: Comprehensive and actionable reporting for decision-making.
- Success Scenario:
 - Team Member enters the application and in the home menu there are options to choose between disease, pest or anomalies reports, after the choose, initiates the report creation process;
 - When making reports, it is necessary to enable the upload of photos taken at the activity location and automatically provide the

GPS coordinates in all instances. The form for diseases and pests should include two comboboxes. One combobox allows the user to select the crop, while the second combobox dynamically displays the disease connected with the selected crop. Additionally, the form should provide the option to make the report either public or private;

- In the context of diseases, when creating a report, the Team Member can utilize the "Automatic Disease Identification" feature by capturing or uploading a snapshot. This photograph is then analyzed and automatically recognized by the app to identify the disease;
- Upon receiving the disease diagnosis, the user has the option to click on "check details" to access the internet and get information regarding the automatically identified ailment;
- After the user submits the form, the app automatically fills in the latitude, longitude, and creation date fields using the GPS capabilities of the mobile device. If all the required values are provided, a notification indicating that the report has been successfully created is displayed;
- On the main page of the app, team members will find a tab bar displaying a list of reports categorized by diseases, pests, and anomalies. Each report will include information such as the report code, mentioned crop, mentioned disease, and the date and time it was created. By clicking on the "see details" button, users can access a page where they can view the specific details of each report.

Manage Team (Website)

This use case is specific to the website, where the Team Member can comprehend their team's structure and retrieve information about team members and the team leader.

- Primary Actor: Team Member
- Preconditions: Team Member must be authenticated with the appropriate privileges.
- Success Scenario:
 - Team Member selects the "Manage Team" option on the website.
 - The system allows the Team Member to "Search" for teams and "See Details", which includes member lists and information about the team creator.
 - Team Member uses these features to gain insights into their team's composition and member roles.

Data Visualization (Web and Mobile)

This use case is present both on the website and in the mobile app for the Team Member. Allows him to have more detailed information about the reports through graphs and geolocation maps.

- Primary Actor: Team Member
- Stakeholders: Team Member, System Analysts, Team Leader
- Preconditions:
 - Website: Team Member must be authenticated with the appropriate privileges.
 - Mobile App: Team Member must be authenticated with the appropriate privileges.
- Guarantee: Comprehensive and actionable reporting for decision-making.
- Success Scenario:
 - Website:

- Team member enters the system, navigates to the reports area and when clicking on one of the reports (disease, pests or anomalies), you will have the option to see the map of the geolocation of the reports and you can click on the option to see detailed information in the graphs;
 - The system displays various forms of data analytics, including interactive charts and graphs.
 - Team Member interacts with the data, gaining insights into different metrics and trends that aid in decision-making processes.
- Mobile App:
 - Team Member accesses the mobile app, and through a navbar can navigate to the Data Visualization page;
 - In the App, you can see on this page a map with the geolocation of the reports carried out by this team member and you can see the calculation of the tasks carried out.

3.4 Storyboards

During the design phase, elaborate storyboards are presented. These visual guides serve as the blueprint for the system's user interface, offering detailed information about each component, such as navigation menus, button placements, page layouts, and user interaction sequences.

An explanation of the interface design process will be given for the web platform to provide a seamless and user-friendly experience. The objective is to ensure that all information is readily available and that operations may be effortlessly executed. This study will focus on the techniques used to improve the user interface of a mobile application for smaller screens, ensuring a consistent user experience across different device sizes.

Upon completion of this part, readers will have a view of the manner in which users will interact with the system, encompassing both web and mobile platforms. Having this comprehension is essential for understanding the reasoning behind our design

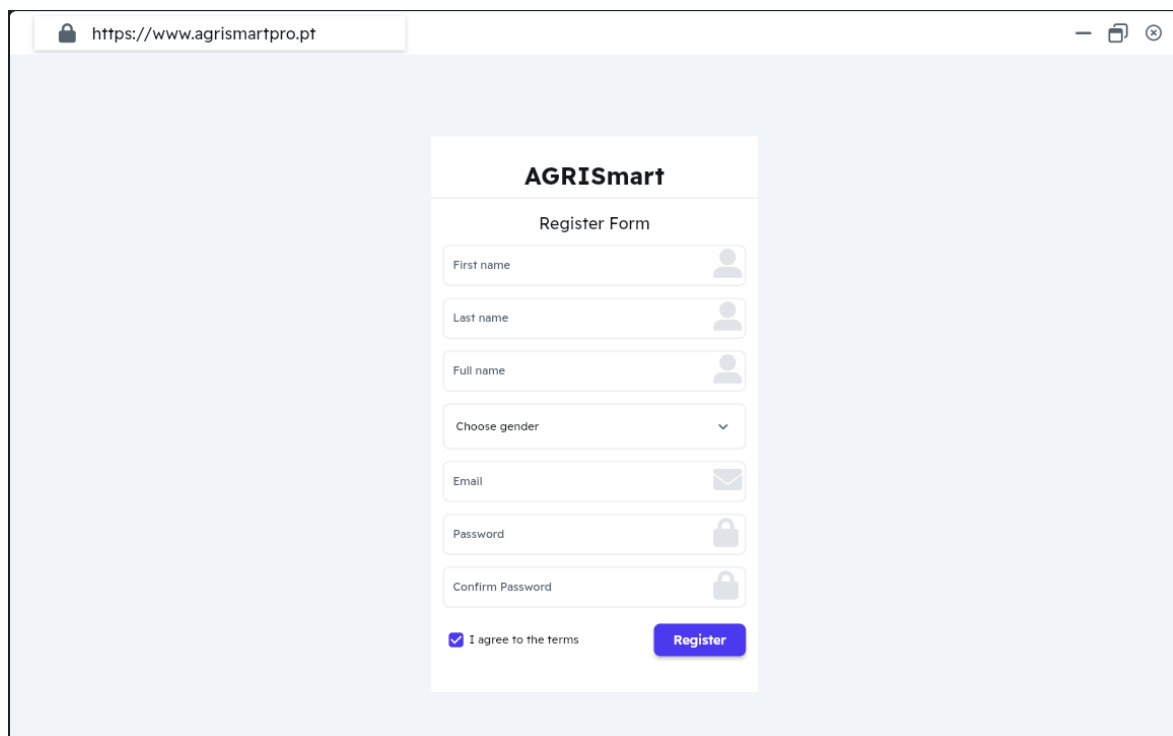
choices and how they contribute to a system that is both user-friendly and efficient. This part demonstrates our dedication to developing a system that is both efficient and user-friendly, providing a seamless and pleasurable user experience.

The following figures illustrate the drawings created in the storyboards for both the Admin Panel and the Mobile App:

Web app - registration screen

Figure 6 depicts the storyboard for the registration screen of the web application page. This page is designated for the registration of users who will serve as Team Leaders on the platform. Users can complete the required fields as indicated by the system.

Figure 6. Storyboard related to the web application registration screen.



The image shows a web browser window with the URL <https://www.agrismartpro.pt>. The main content is a registration form titled "AGRISmart" and "Register Form". The form contains the following fields and elements:

- First name (text input with a person icon)
- Last name (text input with a person icon)
- Full name (text input with a person icon)
- Choose gender (dropdown menu)
- Email (text input with an envelope icon)
- Password (text input with a lock icon)
- Confirm Password (text input with a lock icon)
- A checkbox labeled "I agree to the terms" which is checked.
- A blue "Register" button.

Web app - login screen

Figure 7 depicts the storyboard for the login screen of the online application. This page serves as the entry point for each user to access the platform once they have completed the registration process.

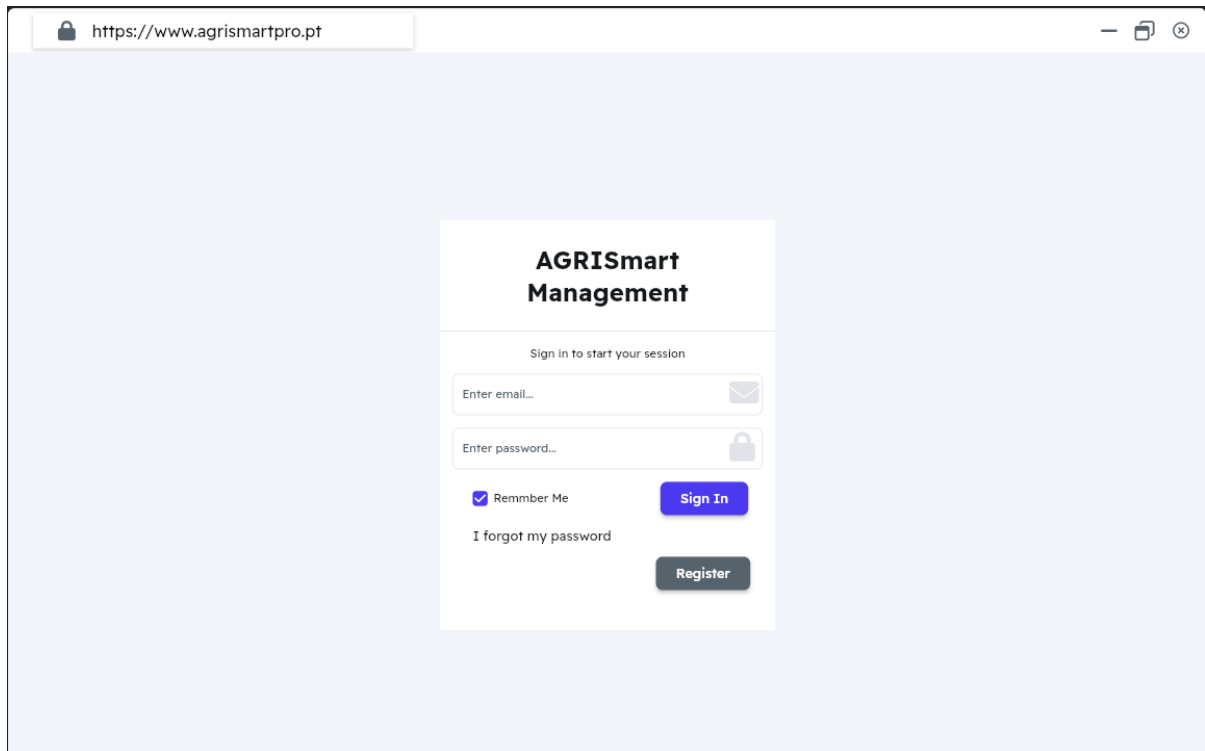


Figure 7. Storyboard related to the web application login screen.

Web app - Admin main screen

Figure 8 displays the storyboard for the web application page, serving as the primary interface for the administrator user. On this page, the administrator effectively oversees the management of the platform's different objects based on the assigned access and rights.

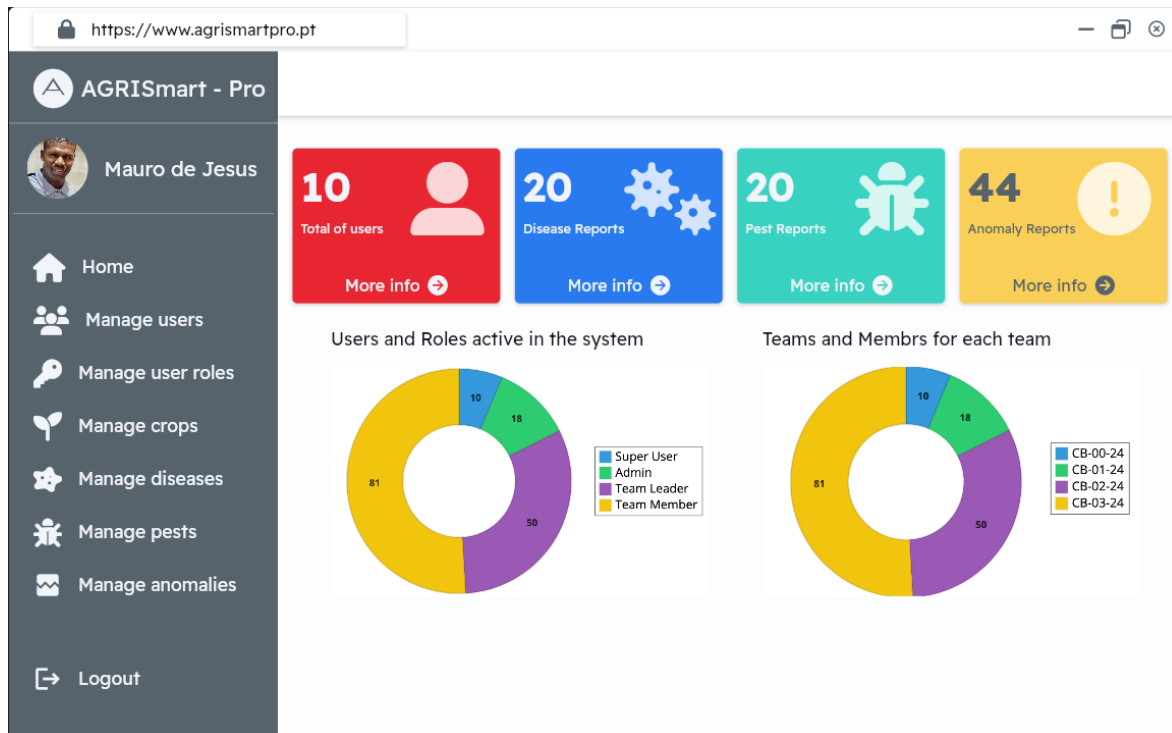


Figure 8. Storyboard related to the web application Admin main screen.

Web app - Team Leader main screen

Figure 9 displays the storyboard for the web application page, serving as the primary interface for users with the role of Team Leader. On this page, the Team Leader effectively oversees the management of the different things on the platform based on the specific access and rights assigned to them.

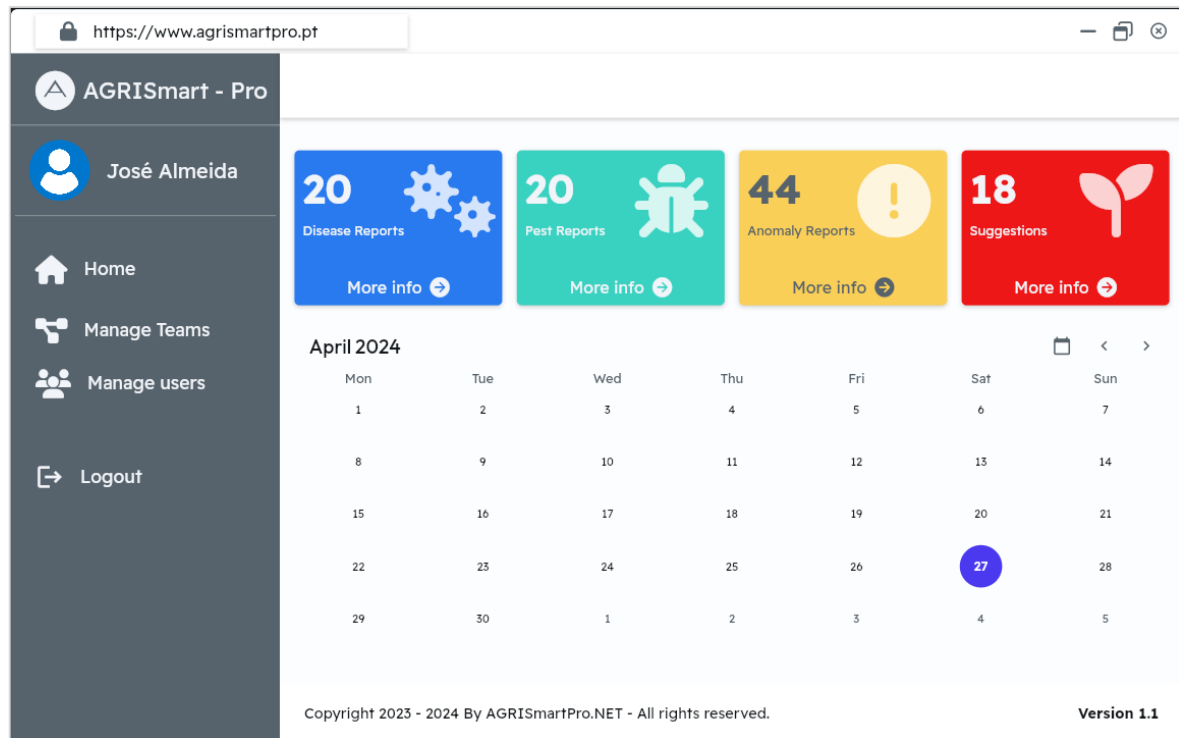


Figure 9. Storyboard related to the web application Team Leader main screen.

Web app - Team Member main screen

Figure 10 depicts the storyboard of the primary screen for the Team Member user in the web application. On this page, the Team Member effectively oversees the management of the platform's different objects based on the specific access and rights assigned to them.

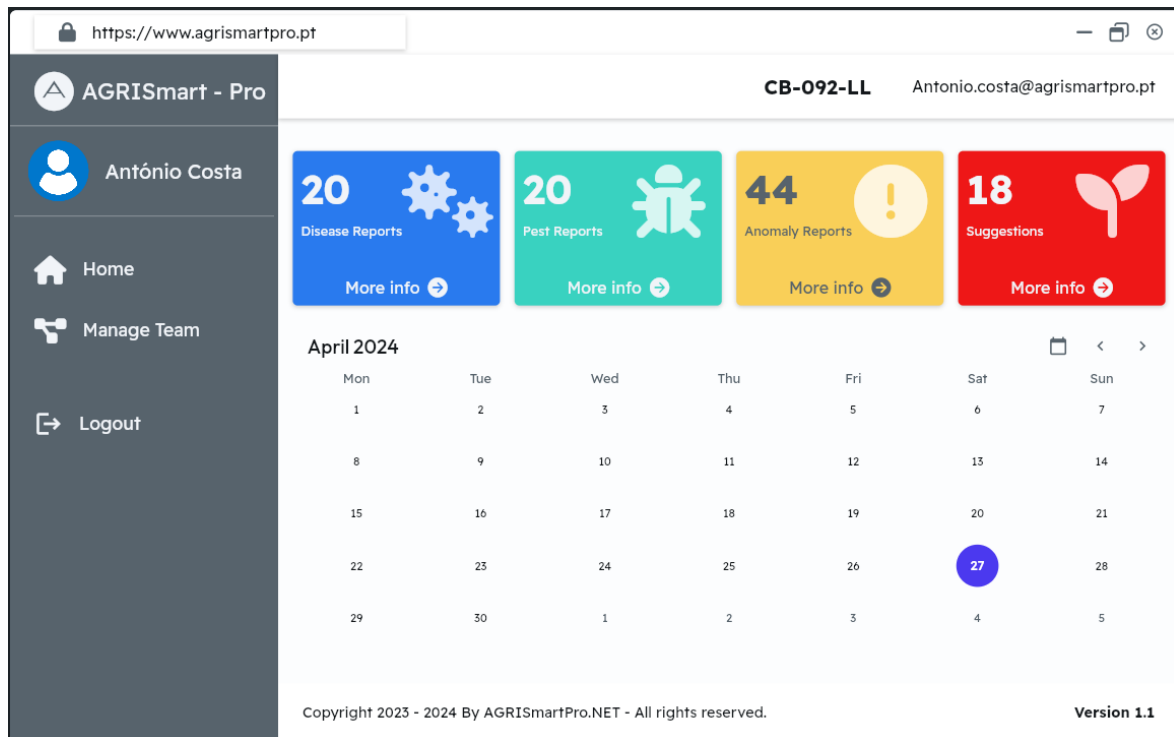


Figure 10. Storyboard related to the web application Team Member main screen.

Web app - Manage Users main screen

Figure 11 depicts the screen for managing users. This screen is restricted to the admin user and enables them to retrieve information on people who have registered on the platform, so that they may effectively oversee and control their actions based on their assigned privileges and permissions.

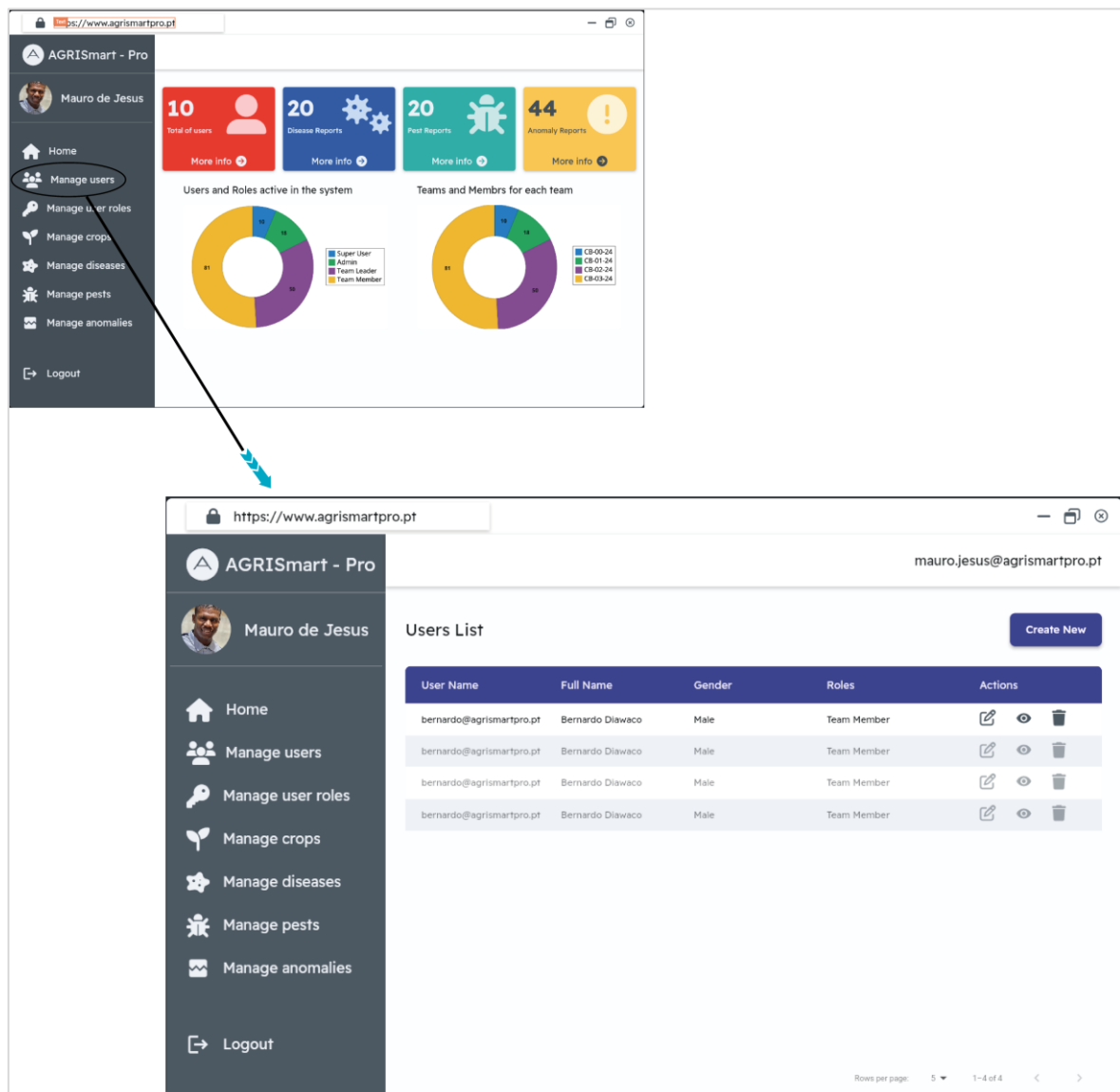


Figure 11. Storyboard related to the web application Manage Users screen.

Web app - Manage Roles main screen

Figure 12 displays the interface for managing user roles. This page is exclusive to the administrator user and grants them access to the roles registered on the platform for the purpose of managing them based on their privileges and permissions.

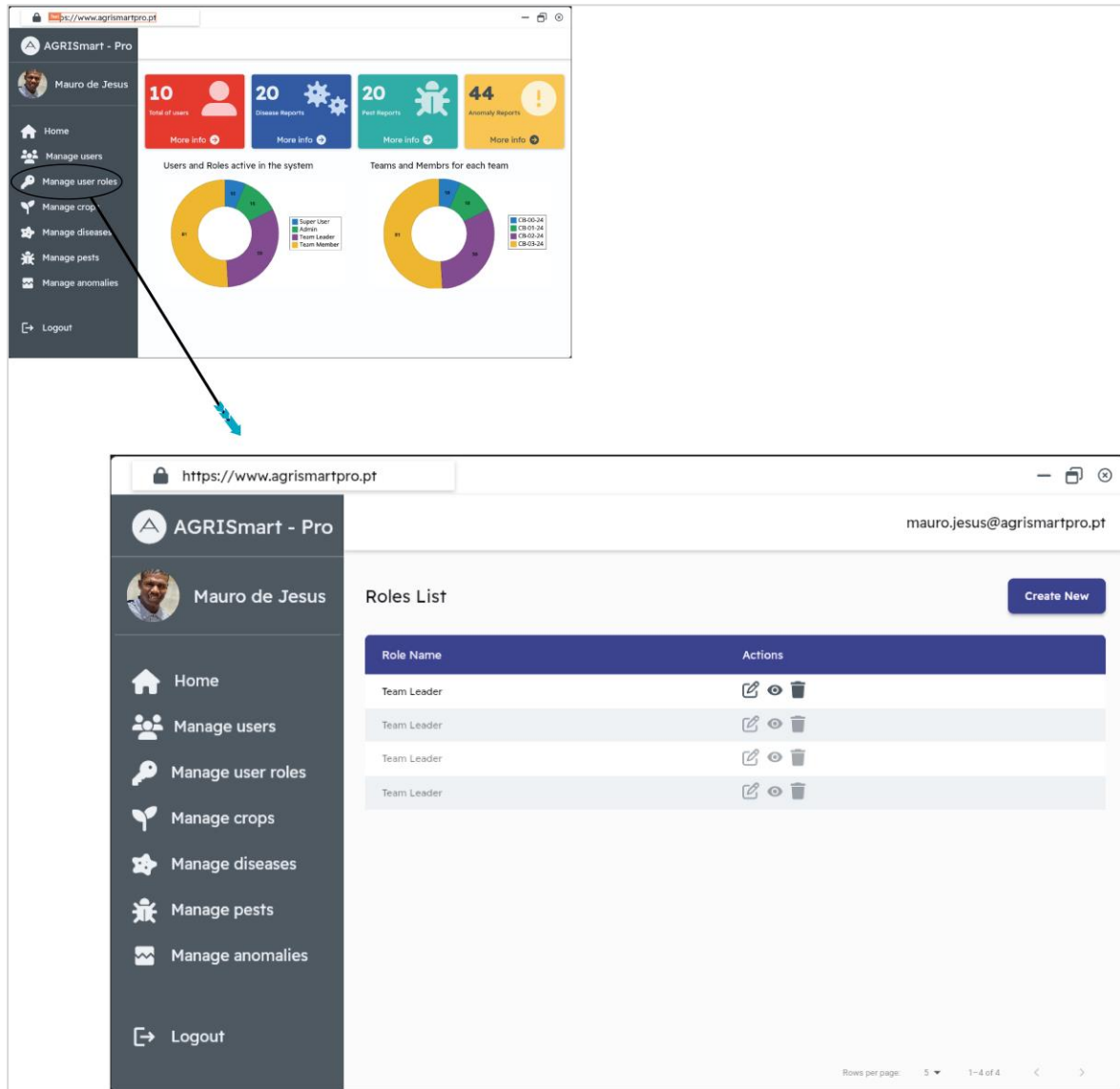


Figure 12. Storyboard related to the web application Manage Roles screen.

Web app - Create new user screen

Figure 13 depicts the user administration pop-up panel, specifically in the context of adding a new user. This page is exclusive to the administrator user and enables them to generate a new user on the platform for the purpose of managing it based on their privileges and permissions.

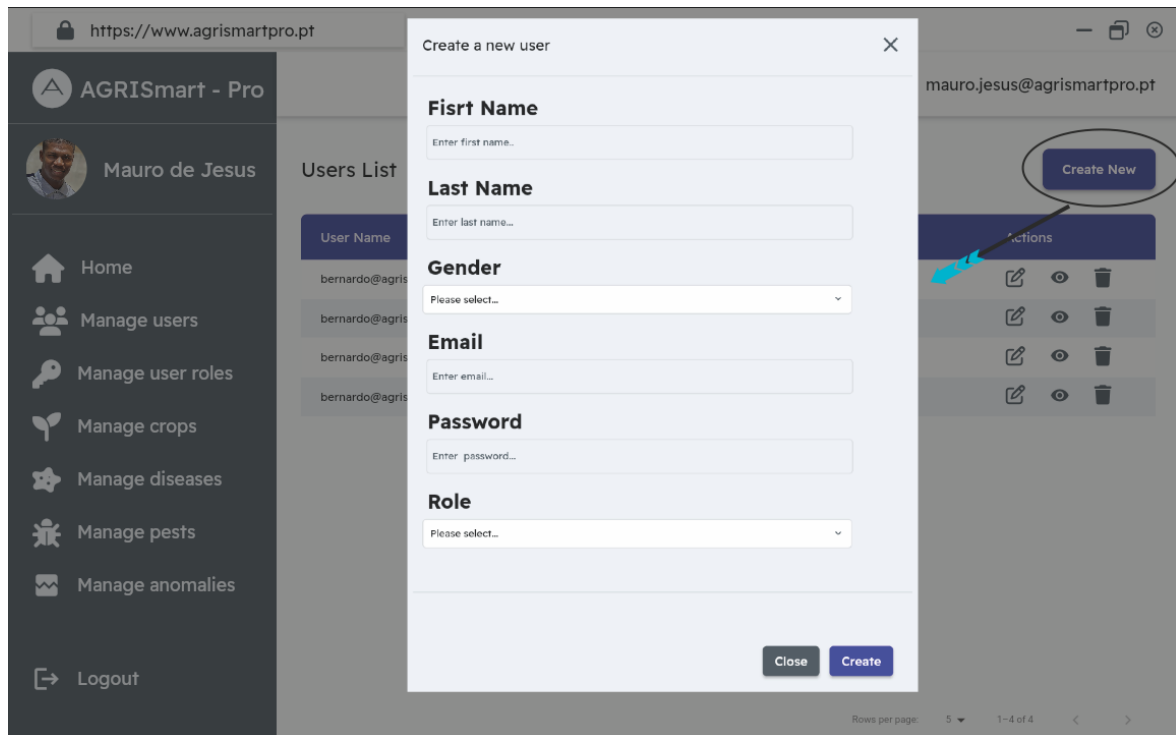


Figure 13. Storyboard related to the web application Create new user screen.

Web app - List and Create new disease report screen

Figure 14 depicts the disease report management screen, specifically displaying the list of reports and the option to create a new report. This screen is accessible only to team leaders and team members. It provides them with the ability to view the existing reports and create a new illness report on the platform. They can then manage these reports based on their specific privileges and permissions.

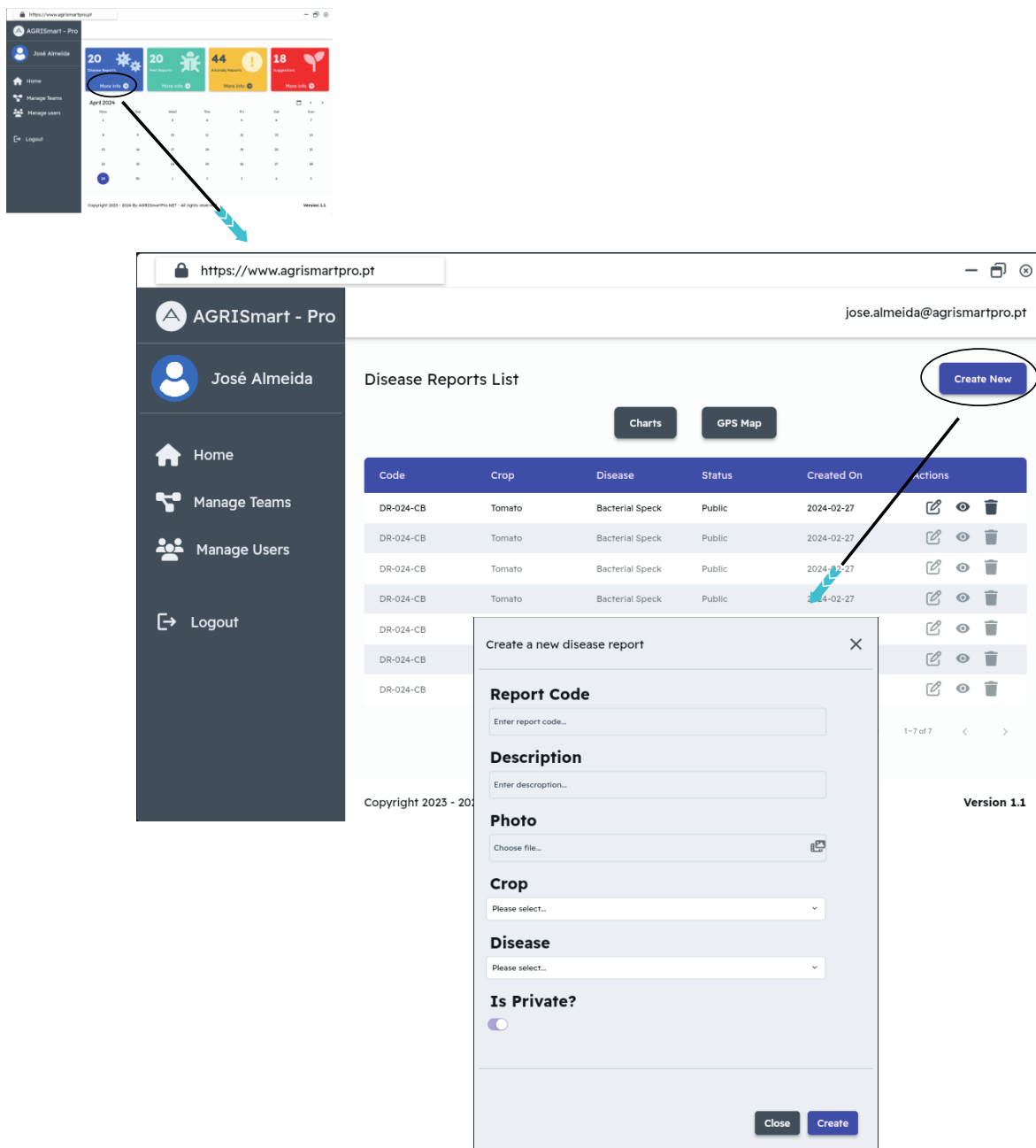


Figure 14. Storyboard related to the web application Create new disease report screen.

Mobile app mixed action screens

Figure 15 presents a concise overview of the operational functionality of the mobile application. This is a visual representation of the actions performed by the App, demonstrating how each item may be accessed through the displays. The picture illustrates a sample of only 6 screens currently being utilized for demonstration purposes. The initial step is encountering the login screen, which grants access to registered users who hold the positions of Team Leader or Team Member, enabling them to enter the system. Following is the primary menu, which provides the operator with many options including: Disease Report (for creating one or multiple disease reports), Pest Reports, Anomaly Reports, and Suggestions Report. On the main screen, there are tabs that allow users to view categorized lists of previously generated reports. Also, users can access detailed information for each report. The next screen displays the particulars of a report, enabling you to get more comprehensive information about it. Displayed below is an illustration of the user interface for generating a Disease Report. It showcases the comprehensive selection of options and fields in accordance with the application's standardization. From the main screen of the App, users can navigate through a menu tab to obtain information like a map displaying the location of reports submitted by the operator, the total number of reports made, and the number of pending tasks.

Basically, there is a representation of how the Mobile App works, which will be explained and shown in more detail using a real-life example.

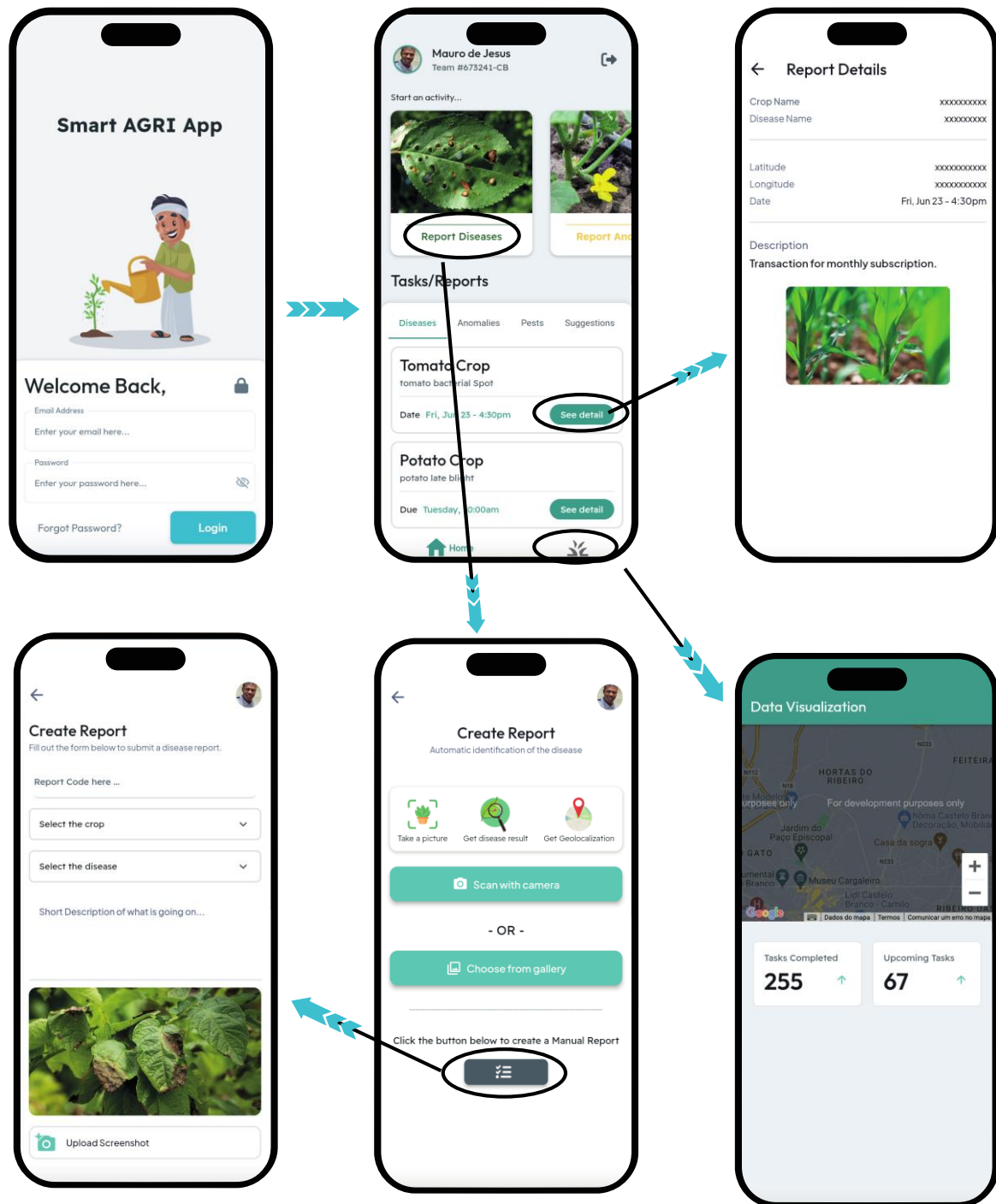


Figure 15. Storyboard related to the mobile application mixed actions screens.

3.5 Data Model Entity-Relationship

This section explores the Entity-Relationship (ER) Data Model, a crucial instrument for modelling and developing databases. The ER model serves to depict the entities inside the problem domain and their interrelationships, offering a broad perspective of the data structure and promoting effective communication and comprehension of the system among stakeholders.

ER model is built on the notions of entities, which represent real-world things or concepts, attributes, which describe features of the entities, and relationships, which define interactions between these entities [87]. This approach is renowned for its capacity to streamline intricate data structures and enhance the clarity of data links, hence facilitating better communication among stakeholders [88].

The ER model method has numerous benefits [88]:

Clarity: Portrays the data system in a lucid and succinct manner, enhancing the comprehension and transmission of data.

Flexibility refers to the ability to adapt and adjust the data system as required, including the addition or removal of entities, properties, or relationships.

Maintenance: Streamlines data system maintenance by presenting a concise and organized depiction of data that enhances the process of identifying and resolving issues.

The conceptual, logical, and physical models created from the ER model are crucial in the requirements analysis process. They serve as the basis for designing the database schema. The ER conceptual model establishes the fundamental reference entities widely employed by the organization, whereas the logical and physical models incorporate operational and transactional specifics, correspondingly.

This chapter will explain the practical use of the ER model through an entity and relationship diagram. This diagram will function as a visual aid in illustrating the project's data structures and delineating the interconnections between them, so aiding the creation of the foundational database and the construction of a cohesive and well-organized information system.

To gain a more thorough comprehension of ER models and their execution, I recommend consulting Peter Chen's research and Visual Paradigm's recommended methods, which provide a complete examination of ER modelling and practical instances of its use [88].

3.5.1 Database Relational Model

After establishing the ER model of the application's database, which captures the interactions between entities and clarifies required and cardinal relationships, a relational model can be constructed with the same level of detail and structure. The primary keys and foreign keys in this relational schema are carefully determined according to the relationships between the tables in the system. In addition, new relational tables are generated to support the intricate relationships between entities specified in the prior ER model. This technique guarantees the exact reflection and management of all relational characteristics within the database.

The subsequent tables form the fundamental structure of the system's database, with each table listed along with its corresponding properties and keys:

- User: A pivotal entity capturing the essential credentials and personal details of the system's participants (attributes: id, first_name, last_name, etc.).
- Team: Encompasses groups of users working in concert within the agricultural ecosystem (attributes: id, name, description, etc.).
- DiseaseReports & PestReports: Crucial for monitoring and managing crop health, these tables register incidents related to crop diseases and pests (attributes: id, code, description, photo, etc.).
- Crop: Represents the variety of crops managed within the application (attributes: id, name, photo, description, etc.).
- AnomalyReports: Facilitates the logging of anomalous events within the agricultural domain (attributes: id, photo, description, etc.).
- SuggestionReports: Table dedicated to capturing recommendations for improving agricultural practices (attributes: id, code, description, etc.).
- TypeIssue: Categorizes various issues encountered within the agricultural process (attribute: name).

The relational model, depicted in the diagram, displays a network of tables connected by primary and foreign keys, representing a strong data governance system necessary for preserving the integrity and usefulness of the agricultural application's data repository. This precise arrangement of tables and relationships creates a strong and adaptable infrastructure that can grow alongside the changing requirements of the application and the dynamic demands of the agriculture sector.

Using this comprehensive approach, Figure 16 visually represents the interconnected framework of the system database. Every table is specifically created to collect vital data and facilitate communication between different elements within the system. The primary and foreign keys were precisely identified to ensure data integrity and appropriately represent the relationships described in the ER model.

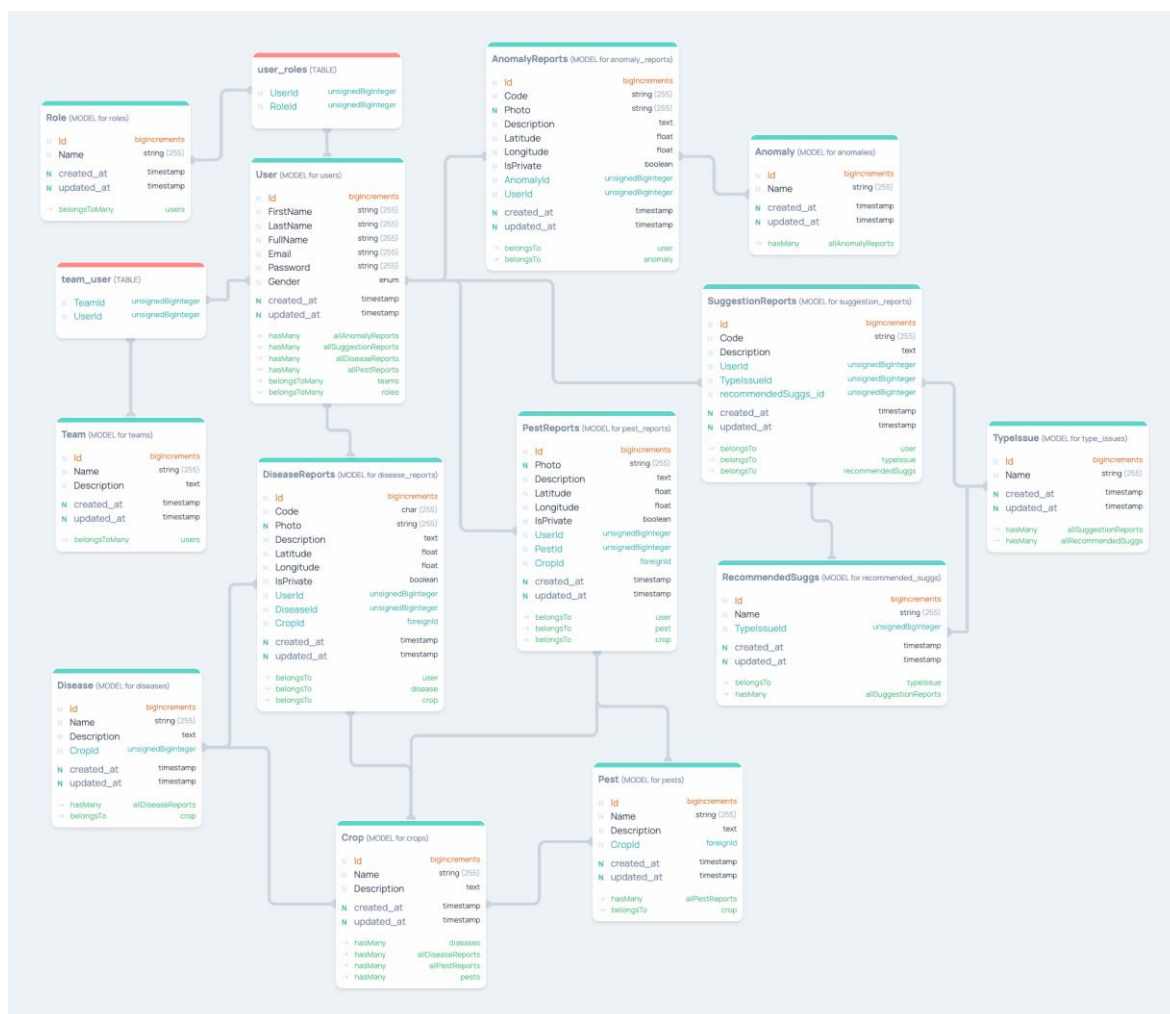


Figure 16. System Data Base Relational Model.

3.5.2 Description of Relational Model Tables

This section provides a detailed description of each table in the system's database model, including the properties and their corresponding data types. A explanation of each attribute is given, including relevant comments that may indicate if the attribute is a primary or foreign key, and notes on the mandatory nature of the data fields.

Thus, the information on the fields in the database that provided the foundation for the administration of these planned agricultural activities is provided in tables 5 through 18.

Table 5. Users Table description.

Field	Data types	Description	Observations
Id	Bigint	User primary key	Not null
FirstName	Varchar (255)	User's first name	Not null
LastName	Varchar (255)	User last name	Not null
Email	Varchar (255)	User email	Not null, unique
Password	Varchar (255)	User password	Not null
Gender	Tinyint	User gender	Not null
CreatlAt	Timestamp	User creation date	Not null
UpdatedAt	Timestamp	User last updated date	Nullable
CreatedByUserId	Nvarchar (128)	User creator	Nullable

Table 6. Roles Table description.

Field	Data types	Description	Observations
Id	Bigint	Role primary key	Not null
Name	Varchar (255)	Role description	Not null, unique
CreatedAt	Timestamp	Role creation date	Not null
UpdatedAt	Timestamp	Role updating date	Nullable

Table 7. UserRoles Table description.

Field	Data types	Description	Observations
UserId	Bigint	User foreign key	Nullable
RoleId	Bigint	Role foreign key	Nullable

Table 8. Teams Table description.

Field	Data types	Description	Observations
Id	Int	Team primary key	Not null
Name	Varchar (255)	Team name	Not null, unique

Table 9. UserTeams Table description.

Field	Data types	Description	Observations
UserId	Bigint	User foreign key	Nullable
TeamId	Int	Team foreign key	Nullable

Table 10. Diseases Table description.

Field	Data types	Description	Observations
Id	Bigint	Disease primary key	Not null
Name	Varchar (255)	Disease name	Not null, unique
Description	Varchar (MAX)	Disease description	Nullable
CropId	Int	Crop foreign key	Not null

Table 11. Disease Reports Table description.

Field	Data types	Description	Observations
Id	Bigint	Disease report primary key	Not null
Code	Varchar (255)	Disease report code	Not null, unique
CropId	Int	Crop foreign key	Not null
DiseaseId	Int	Disease foreign key	
Description	Varchar (MAX)	Disease description	Nullable
Photo	Varchar (255)	Report photo	Nullable
Latitude	Float	Report GPS	Not null
Longitude	Float	Report GPS	Not null
UserId	Int	Report creator	Not null
CreatedAt	Timestamp	Report creation date and time	Not null
UpdatedAt	Timestamp	Report updating date and time	Nullable

Table 12. Pests Table description

Field	Data types	Description	Observations
Id	Bigint	Pest primary key	Not null
Name	Varchar (255)	Pest name	Not null, unique
Description	Varchar (MAX)	Pest description	Nullable
CropId	Int	Pest foreign key	Not null

Table 13. Pest Reports Table description.

Field	Data types	Description	Observations
Id	Bigint	Pest report primary key	Not null
Code	Varchar (255)	Pest report code	Not null, unique
CropId	Int	Crop foreign key	Not null
PestId	Int	Pest foreign key	
Description	Varchar (MAX)	Pest description	Nullable
Photo	Varchar (255)	Pest report photo	Nullable
Latitude	Float	Report GPS	Not null
Longitude	Float	Report GPS	Not null
UserId	Int	Report creator	Not null
CreatedAt	Timestamp	Report creation date and time	Not null
UpdatedAt	Timestamp	Report updating date and time	Nullable

Table 14. Anomalies Table description.

Field	Data types	Description	Observations
Id	Bigint	Anomaly primary key	Not null
Name	Varchar (255)	Anomaly name	Not null, unique
CreatedAt	Timestamp	Anomaly creation date and time	Nullable
UpdatedAt	Timestamp	Anomaly updating date and time	Not null

Table 15. Anomaly Reports Table description.

Field	Data types	Description	Observations
Id	Bigint	Anomaly report primary key	Not null
Code	Varchar (255)	Pest report code	Not null, unique
Description	Varchar (MAX)	Pest description	Nullable
Photo	Varchar (255)	Pest repot photo	Nullable
Latitude	Float	Report GPS	Not null
Longitude	Float	Report GPS	Not null
UserId	Int	Report creator	Not null
CreatedAt	Timestamp	Report creation date and time	Not null
UpdatedAt	Timestamp	Report updating date and time	Nullable

Table 16. Type Issues Table description

Field	Data types	Description	Observations
Id	Bigint	Type issue primary key	Not null
Name	Varchar (255)	Issue name	Not null, unique
CreatedAt	Timestamp	Issue creation date and time	Nullable
UpdatedAt	Timestamp	Issue updating date and time	Not null

Table 17. Recommended suggestions Table description

Field	Data types	Description	Observations
Id	Bigint	Recommended suggestions primary key	Not null
Name	Varchar (255)	Suggestions name	Not null, unique
TypeIssueId	Int	Type issue foreign key	Not null
CreatedAt	Timestamp	Issue creation date and time	Nullable
UpdatedAt	Timestamp	Issue updating date and time	Not null

Table 18. Suggestion Reports Table description

Field	Data types	Description	Observations
Id	Bigint	Suggestion report primary key	Not null
Code	Varchar (255)	Report code	Not null, unique
Description	Varchar (MAX)	Report description	Nullable
TypeIssueId	Int	Type issue foreign key	Not null
RecommendedSuggsId	Id	Recommended suggestions foreign key	Not null
UserId	Id	Created by UserId	Not null
CreatedAt	Timestamp	Issue creation date and time	Nullable
UpdatedAt	Timestamp	Issue updating date and time	Not null

3.6 Final Remarks

The emphasis of this chapter was on outlining the functional and non-functional requirements, necessary to create a system that can geolocate and diagnose diseases, pests, and anomalies in agricultural crops. The chapter started by classifying these requirements and stressing how important they are to software development. Functional requirements specify the operations and features the system must carry out, such data processing, user authentication, and reporting. Conversely, operational requirements specify the system's performance metrics, dependability, security, and usability.

Use case diagrams were essential in graphically illustrating these requirements. The explanations of every use case gave a thorough picture of the relationships among the many system players—Admins, Team Leaders, and Team Members. A user-friendly experience was guaranteed by storyboards, which further illustrated the user interface design.

Emphasizing its significance in database modeling and development, the chapter also covered the ER Data Model. Clear, adaptable, and maintainable, the ER model is an invaluable tool for illustrating data structures and interactions within the system. All things considered, this chapter offered a comprehensive examination of the specifications and modeling procedures needed to build a reliable and efficient system that satisfies user expectations as well as project objectives.

4. Architecture, Tools and Implementations

This chapter provides a presentation of the system's architecture, the tools utilized in its development, and the implementation methods adopted to successfully complete the project. The chapter is structured into three main sections.

Section 4.1 provides an overview of the system's structure and architecture. The description pertains to the modular and scalable design that improves the system's maintainability and efficiency in inter-component communication. The system's fundamental component is a RESTful web service hosted on a dependable web server. This web service acts as an intermediary between the database and client applications, providing dynamic functionality. This design enables effortless integration with diverse client interfaces, such as mobile applications, desktop software and web browsers.

Section 4.2 presents the tools used for web and mobile application development, explaining their roles and how they contribute to the overall operation of the system. The implementation involves utilizing the ASP.NET MVC framework for the Admin Panel, employing the C# programming language, and opting for Flutter and Dart for mobile application development. The section also addresses the incorporation of machine learning models to enhance picture identification capabilities, which are crucial for promptly identifying diseases and pests in agricultural crops.

Section 4.3 presents some techniques employed in creating the Admin Panel and the Mobile App. The content elucidates the practical application of tools and architectural design choices in constructing a resilient, efficient, and user-friendly system. The section aims to provide readers with a clear and thorough understanding of the step-by-step approach used to implement the different components of the system.

Upon completing this chapter, readers will possess a thorough comprehension of the system's design and implementation, gaining a deep appreciation for the complex interaction among its different components and the reasoning behind the selected design choices.

4.1 System Architecture

The solution was developed based on the architecture depicted in the image below, Figure 17, which features a contemporary and modular approach that enhances scalability, maintainability, and efficiency in inter-application and inter-platform communication.

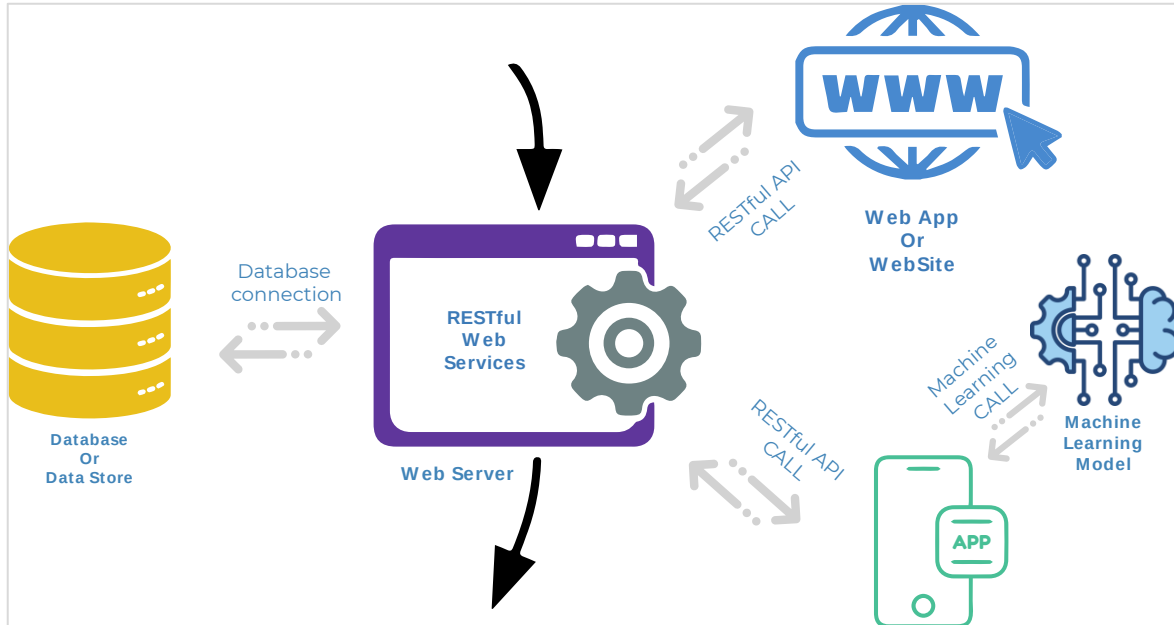


Figure 17. System Architecture.

The core of the system is a RESTful web service, which is hosted on a reliable web server. This server acts as a dynamic intermediary between the database or data storage and the client applications. This vital service efficiently handles requests and responses, utilizing a standardized format, primarily JSON or XML, transmitted via the HTTP protocol. By using a RESTful API design, it becomes possible to seamlessly and flexibly integrate with different client interfaces, including mobile apps, desktop programs, and web browsers. Furthermore, it enhances the system's modular methodology.

By using this architecture, it guarantees that any changes made to the business logic of the web service are deployed without directly affecting end-users. This design paradigm facilitates the autonomous development, testing, deployment, and scaling of each individual system component. A modular system simplifies maintenance, enables

updates, and efficiently accommodates a growing number of users and data requirements.

The framework enhances security by implementing a web service that acts as a protective layer, preventing direct external access to the database. This effectively reduces the vulnerability to cyber-attacks. This configuration not only ensures the protection of data but also streamlines the process of transitioning between databases, due to the standardized communication protocols established between the database and the web service.

One significant improvement in architecture is the incorporation of Machine Learning. This feature encompasses a complex model that has undergone significant training utilizing a wide range of images. The model is stored within a machine learning training environment. This model undergoes regular refinement and updates to improve its accuracy and versatility. After being trained, the model is packaged into a deployable file that can be easily accessed through our mobile application. This method enables the mobile app to utilize advanced picture recognition capabilities, permitting automatic and efficient detection of features. This is crucial for providing users with real-time insights and responses.

Ultimately, this architectural design facilitates exceptional compatibility, allowing various apps to simply connect via well-established web protocols. This enables both internal integration and external system extension, allowing for the changing requirements of the project and the wider market demands. The system architecture is strong and flexible, guaranteeing that the solution will last and be able to handle growth and changes in both project goals and market conditions.

4.2 Tools

This section will mention the tools employed in the development of the system reported in this study. Both the web and mobile components will be shown in a manner that offers an understanding of the tools employed, thereby effectively conveying the purpose of each.

4.2.1 Admin Panel

The ASP.NET MVC framework was selected for the construction of the Administration Panel (Web system), together with the C# programming language. This combination provides a strong and well-developed framework for creating online applications, utilizing the broad functionality of the .NET toolkit and the clarity of C#. C# is known for its sophisticated features such as LINQ (Language Integrated Query) and asynchronous programming [89].

ASP.NET Identity for Authentication:

User authentication is a crucial component of our system and was used ASP.NET Identity to implement it. This framework facilitates authentication through cookies, offering a secure and efficient approach to managing user sessions on the online system and mobile application.

Entity Framework 6 (EF6):

The EF6 framework was utilized with the Database First methodology to develop models based on an existing database schema. This strategy expedites development and minimizes inconsistencies between database architecture and application code. EF6 is an advanced Object-Relational Mapping (ORM) framework that simplifies database interactions. It enables developers to work with strongly typed C# objects at a higher level of abstraction [90].

SQL Server and Visual Studio 2022:

Integrating with SQL Server allows for the utilization of its dependable and effective capabilities as a database management system, which is crucial for enterprise-level applications that require utmost data integrity and fast transaction processing speed

[91]. The process of development was expedited by utilizing Microsoft Visual Studio 2022, a widely accepted Integrated Development Environment (IDE) for .NET applications. This IDE offers a smooth interface with ASP.NET MVC and SQL Server, which are commonly used technologies in the industry.

RESTful API and Tests with Postman:

It was created a Representational State Transfer (REST) Application Programming Interface (API) within our Admin Panel framework. To verify the proper functionality of our API prior to its launch, was employed Postman as a tool to test the endpoints. The Postman tool is highly potent, enabling developers to efficiently generate, distribute, and evaluate APIs, guaranteeing their resilience and security.

4.2.2 Mobile App

This section presents a summary of the main technologies and approaches used in creating the mobile application for our system. The subsequent parts provide an explanation of how Flutter and Dart are used for app development, the selection of VS Code as the preferred development tool, the integration of a RESTful API for communication, and the inclusion of machine learning capabilities using Google's Teachable Machine. The selection of these components was made to ensure that the program possesses superior quality, effectiveness, security, and the ability to adapt to future improvements.

Flutter and Dart

Flutter, which utilizes the Dart programming language, was selected for the mobile application's development. Flutter is a user interface development kit developed by Google that is open source. It is known for its ability to create high-quality applications for both Android and iOS platforms using a single source code [92]. This accelerates the development process and streamlines maintenance.

VS Code and API Consumption

VS Code was utilized for the development of the mobile application, a decision renowned for its lightweight nature and wide customization options [93]. The mobile application utilizes the RESTful API created for the Administration Panel, employing

safe protocols for HTTP-based communications, guaranteeing efficient and secure integration across platforms.

Machine Learning Integration with Google's Teachable Machine

The incorporation of a machine learning model built using Google's Teachable Machine simplifies the process of training models for picture categorization tasks within the app, eliminating the need for extensive understanding of machine learning algorithms. The model was exported in TensorFlow Lite format, which is optimized for running on mobile devices. It provides fast processing speed and has a small memory usage [94].

Throughout the development process, best practices and industry standards were followed to guarantee that our system is not only operational, but also highly secure, adaptable, and simple to maintain. This strategy involves not just the development of code, but also the robustness of the system architecture, its capacity to adapt to future requirements, and its preparedness to scale with technological advancements.

4.3 Implementation

This section demonstrates the actual use of the previously outlined concepts and objectives in the creation of the web and mobile system. The system is comprised of two primary components: the Admin Panel and the Mobile Application.

4.3.1 Admin Panel

The practical implementation of the Admin Panel commenced with the configuration of Microsoft Identity, an API that offers capabilities for user authentication, registration, role management, and more.

In this project, Microsoft Identity was employed to oversee user authentication and authorization. This process entailed the configuration of the Identity middleware within the Startup.cs file and the establishment of the requisite database schema utilizing Entity Framework Code First Migrations.

Upon the successful setup of the Microsoft Identity, its features were leveraged to implement user management functionality within the administration panel. This encompassed the creation, editing, and deletion of user accounts, the assignment of roles, and the management of user permissions.

Figure 18 depicts the initialization of Admin Panel and a practical illustration of the aforementioned concept, specifically involving the utilization of Microsoft Identity for the purpose of establishing access levels and validating users inside the system. Here, a person is attempting to enter the system using incorrect credentials, and the application responds with an alert about the said error.

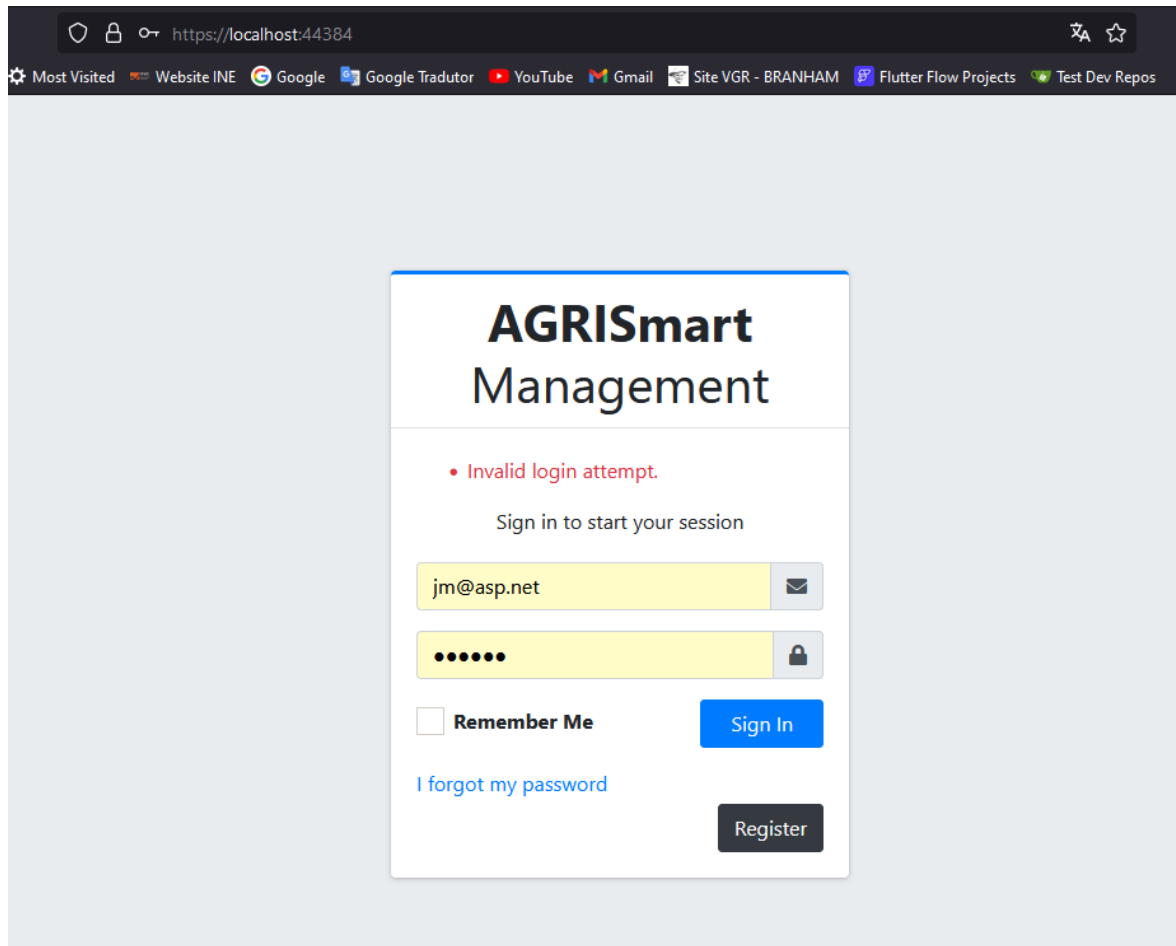


Figure 18. Admin Panel startup screen – Login.

Figure 19 depicts the realization of the concepts presented in the storyboards about the User Admin main page. This is a demonstration of how graphs are displayed. One graph shows users categorized by their roles, while another graph shows the teams in the system along with the number of members in each team.

The outcome of the aforementioned implementation, which is built upon the system architecture, is intended to provide a clear representation of how the data will be shown in the system.

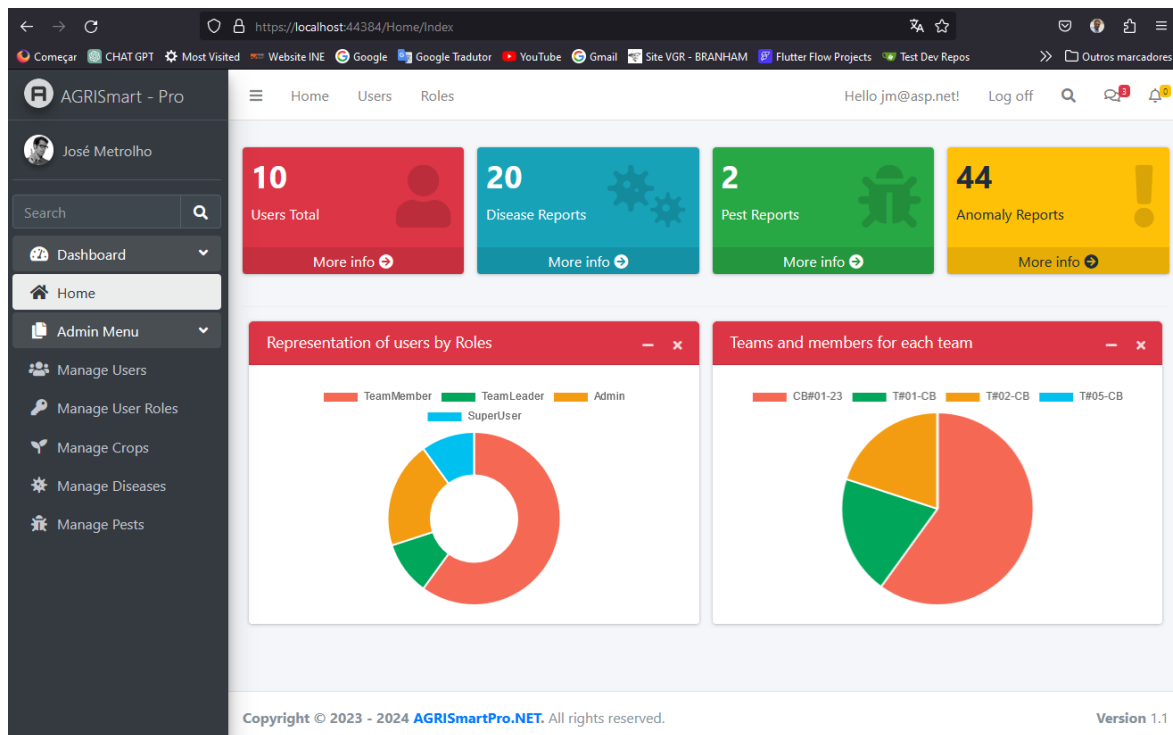


Figure 19. Admin Panel | admin user home screen.

Figure 20 presents the actual implementation of the user management system as described in the use cases and shown in the storyboards. All the actions occurring in the system are simultaneously recorded for the purpose of illustration. The diagram displays a roster of users and the CRUD actions, including an illustration of user creation, concluding with a success message that confirms the successful completion of the user creation process in the system.

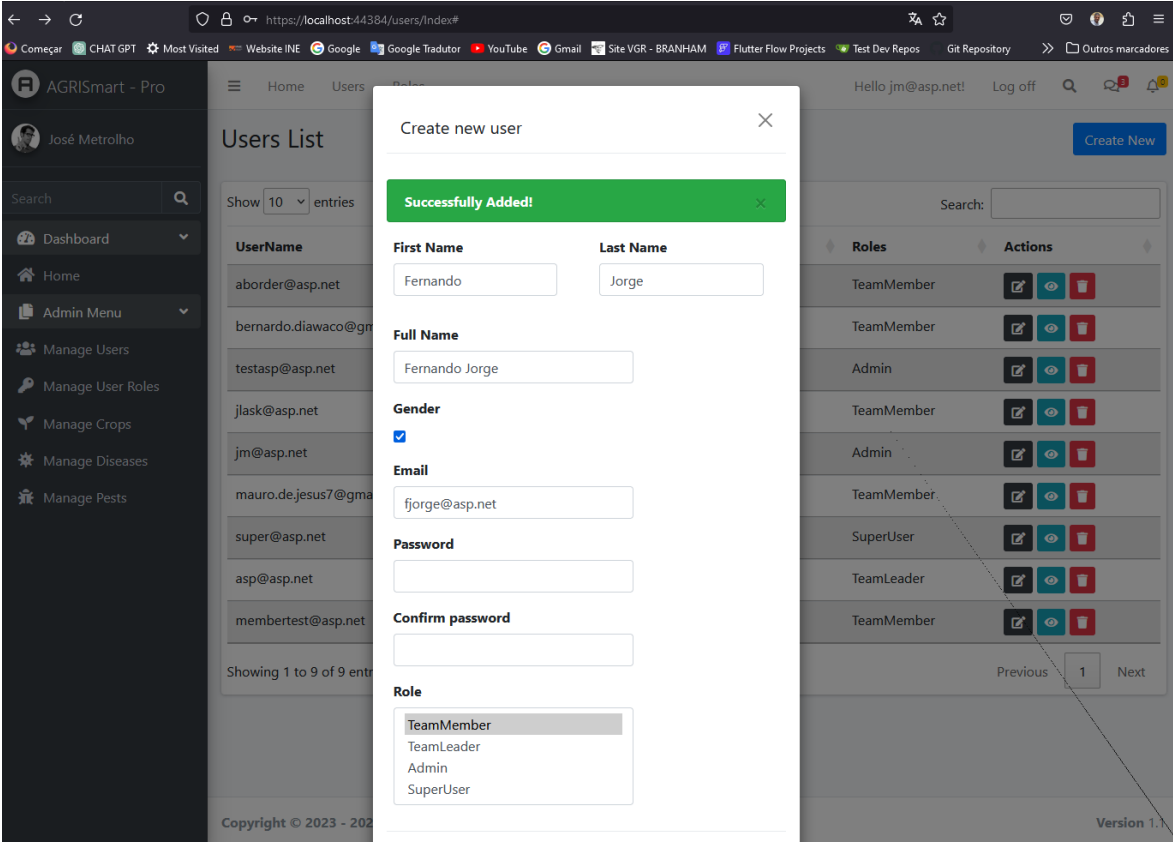


Figure 20. Admin Panel | admin user panel – manage users.

Figure 21 illustrates the result of incorporating the features of the system's user role management area, which is identical to what was depicted in the previous image. Similarly, a roster is displayed that displays four pre-existing roles. The system provides options for doing CRUD tasks. A demonstration of the create action may be watched, along with a success message. This efficiently demonstrates the utilization of data in various access scenarios.

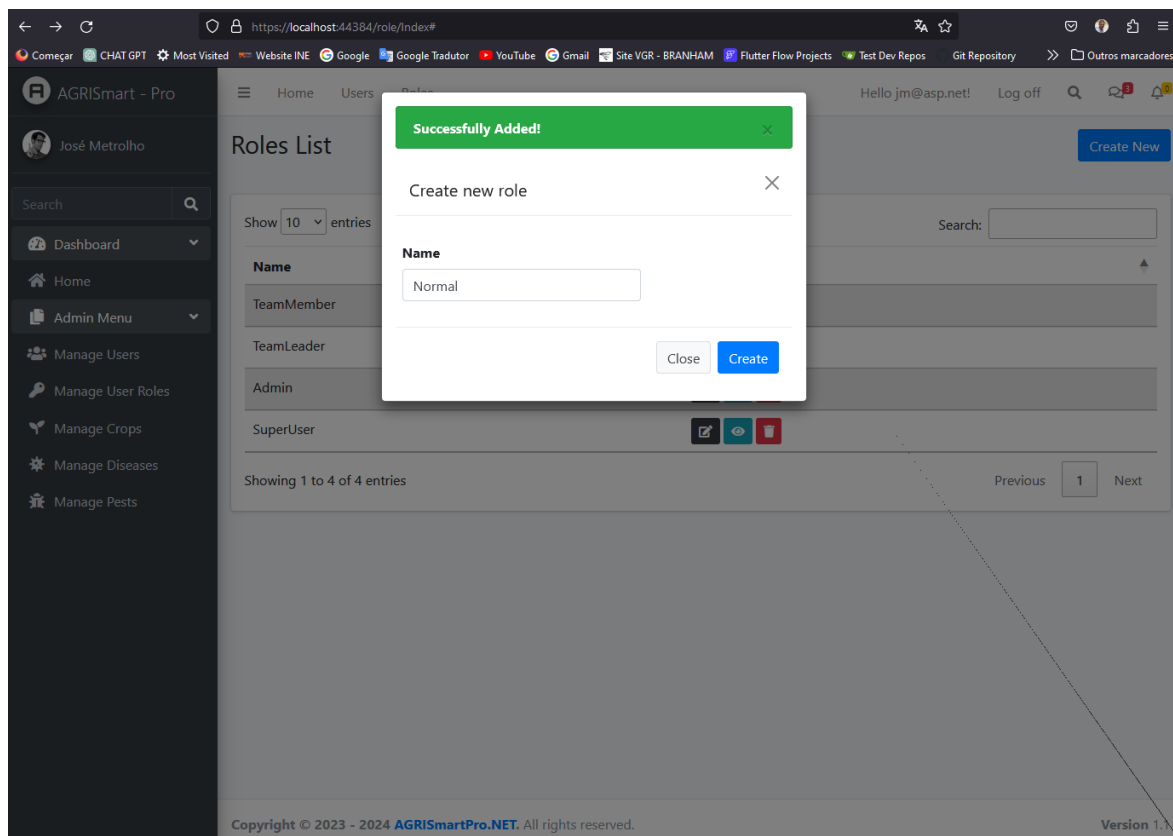


Figure 21. Admin Panel | admin user panel – manage roles.

The Admin Panel has been created with the fundamental features of CRUD operations for managing users, roles, crops, illnesses, pests, and anomalies. These actions employ a consistent strategy and aim to enhance the user's experience by making the system more suitable for agricultural and related sectors.

The following text will demonstrate the outcome of the Team Leader's implementation of management. It will display information that was indicated in the use cases and is now made possible by the application of tools. This demonstration presents the various features and utilities of the administration panel, also known as the Admin Panel.

Figure 22 depicts the realization of the Admin Panel designed specifically for the Team Leader role. The approaches described in the use cases were translated using the employed technologies. The system offers menus for different actions, including team management, member management, reports, and a calendar. The calendar allows users to view the creation dates of each report. In the provided example, reports created in February 2024 are displayed. Different colors are used to differentiate between types of reports, with blue representing Disease Reports (DR) and green representing Pest Reports (PR). As previously mentioned, in this calendar, the team leader has access to public reports, reports made by themselves, and reports created by their field agents.

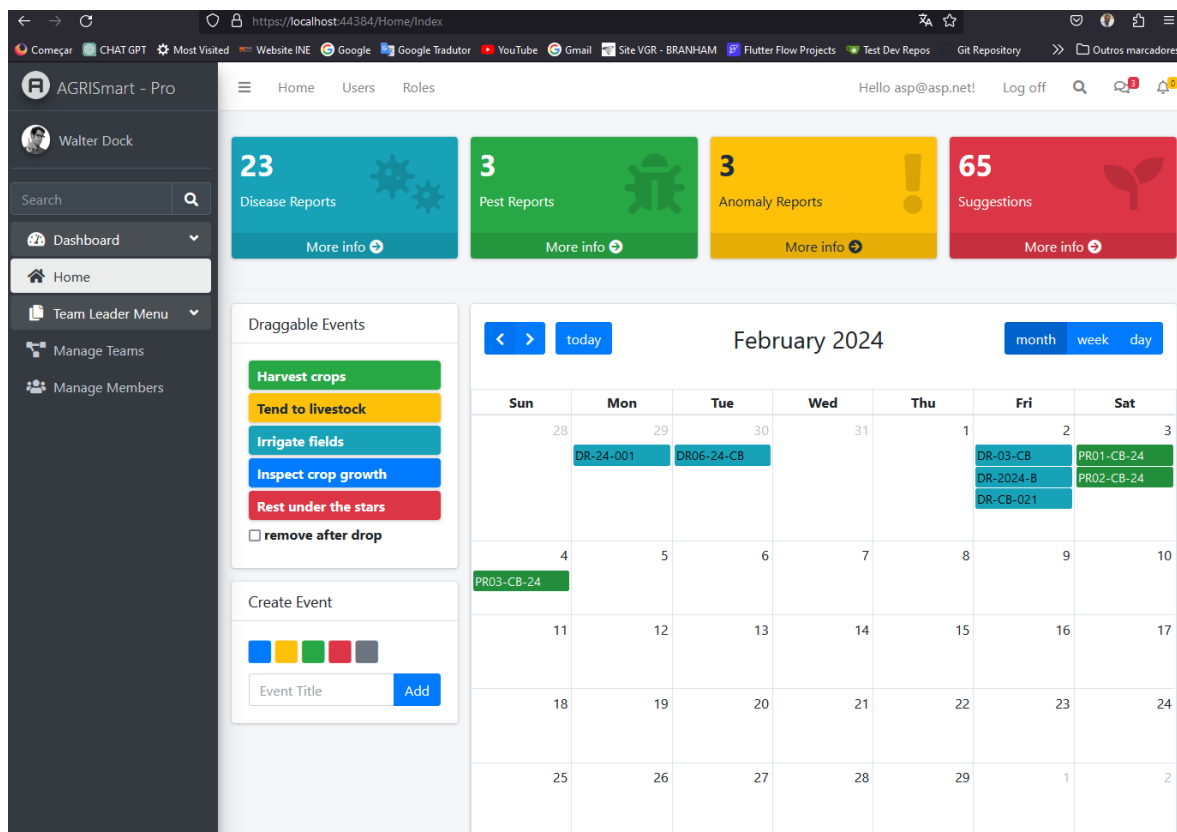


Figure 22. Admin Panel | team leader panel – main screen.

Figure 23 depicts the tangible execution of Teams management. The graphic illustrates the Teams administration, which includes a list of Teams and the CRUD operations, as previously mentioned in the use cases. The lowest most feature is the yellow button that enables you to dispatch an invitation link to a member, inviting them to join your team. The picture below illustrates two separate scenarios. The first scenario demonstrates the editing interface where team members can be added. The second scenario depicts the process of sending links to enable users to join the team.

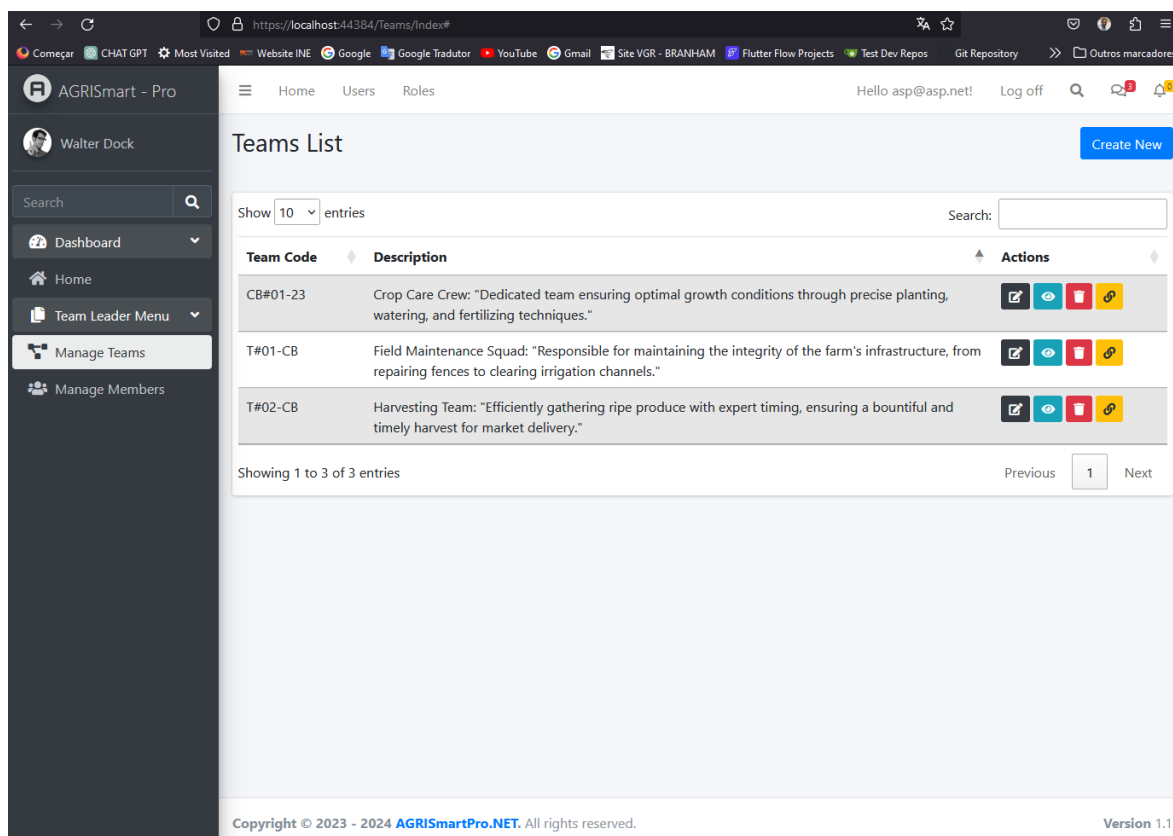


Figure 23. Admin Panel | team leader panel – teams management.

Figures 24 and 25 provide a visual representation of the previously described details. In Figure 24, users may access the modal containing the Team details for editing by clicking on the black "Edit" icon. Within this modal, there is a specific section where users can add new members to the team. In this scenario, there are a total of 4 persons who are eligible, out of which 3 are currently part of the team under consideration.

Figure 26 demonstrates the process of sending a link to a specific member. By clicking on the yellow button with the hyperlink icon, a modal window opens where you can enter your email and send the invitation. The member will then receive an email with a link that redirects them to a registration form. Upon registering, they will be directly added to the team led by the team leader who sent the invitation.

After examining these two figures, it will be possible to observe the favorable result that occurs when the link is sent and the email is received by the member.

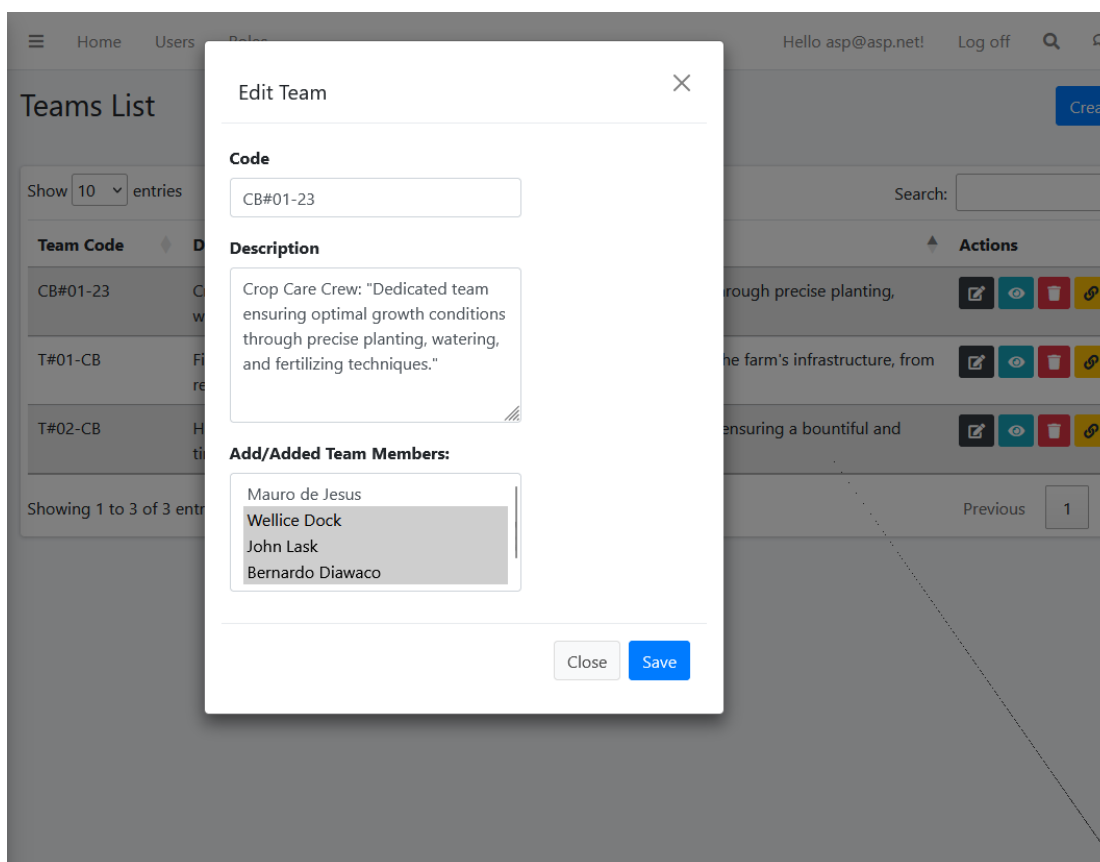


Figure 24. Admin Panel | team leader panel – teams management – edit team

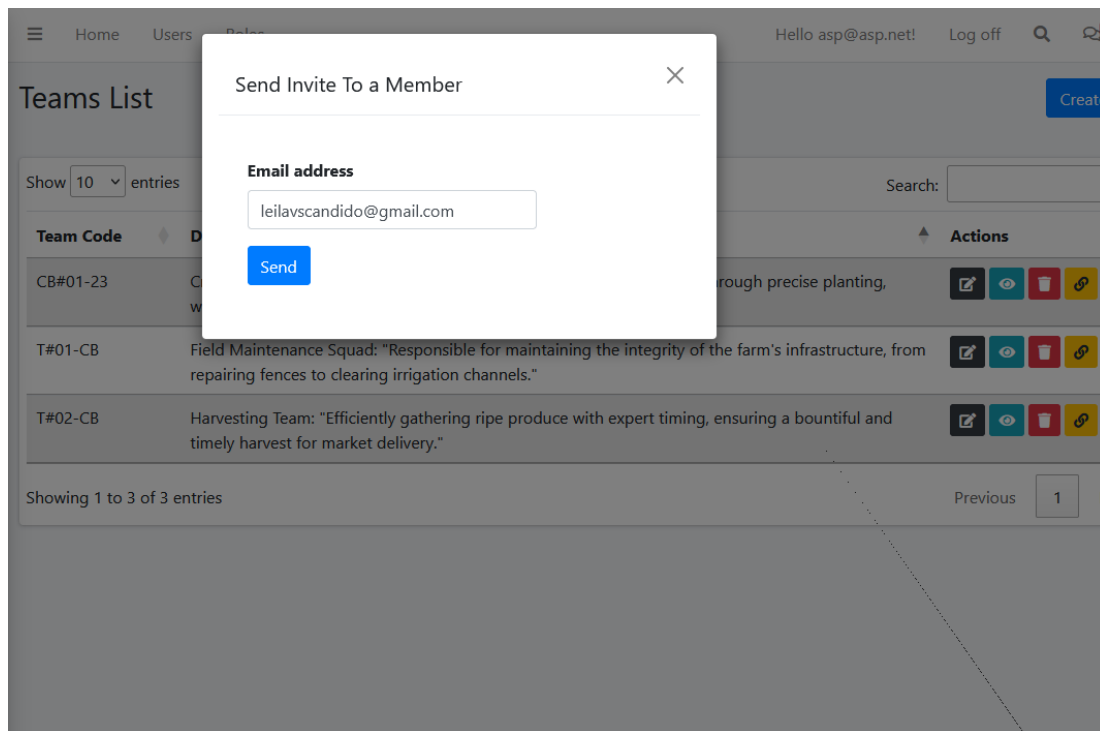


Figure 25. Admin Panel | team leader panel – teams management – send invite

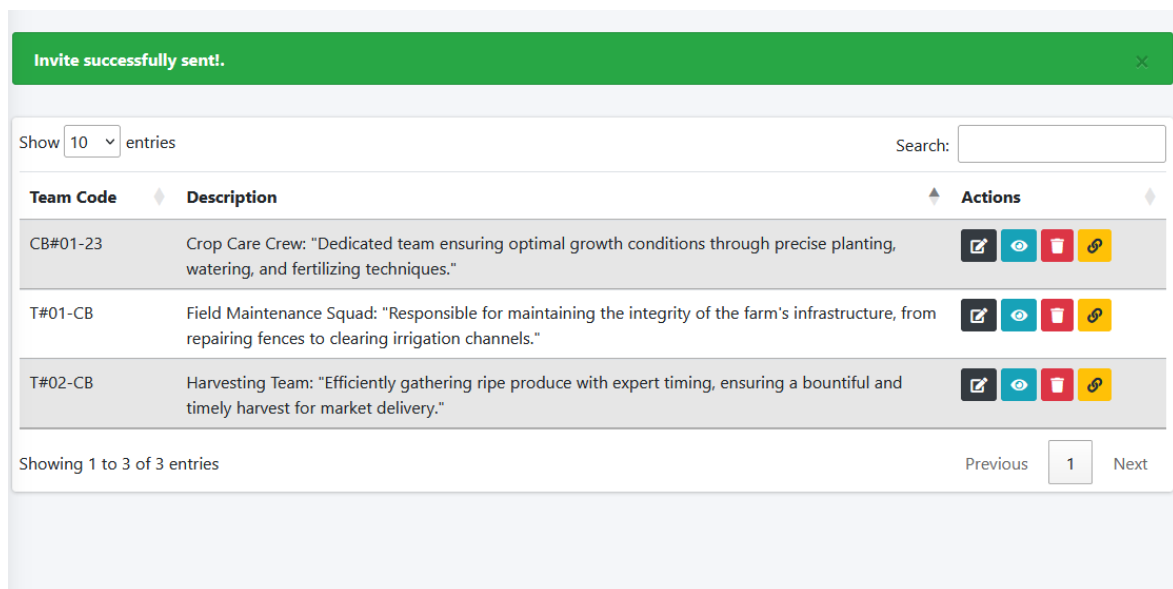


Figure 26. Admin Panel | team leader panel – teams management – send invite successful message

Figure 27 illustrates that the recipient of this invitation is directed to the registration page upon clicking it. Upon registration, user will have the opportunity to join one of the teams overseen by the team leader who extended the invitation.

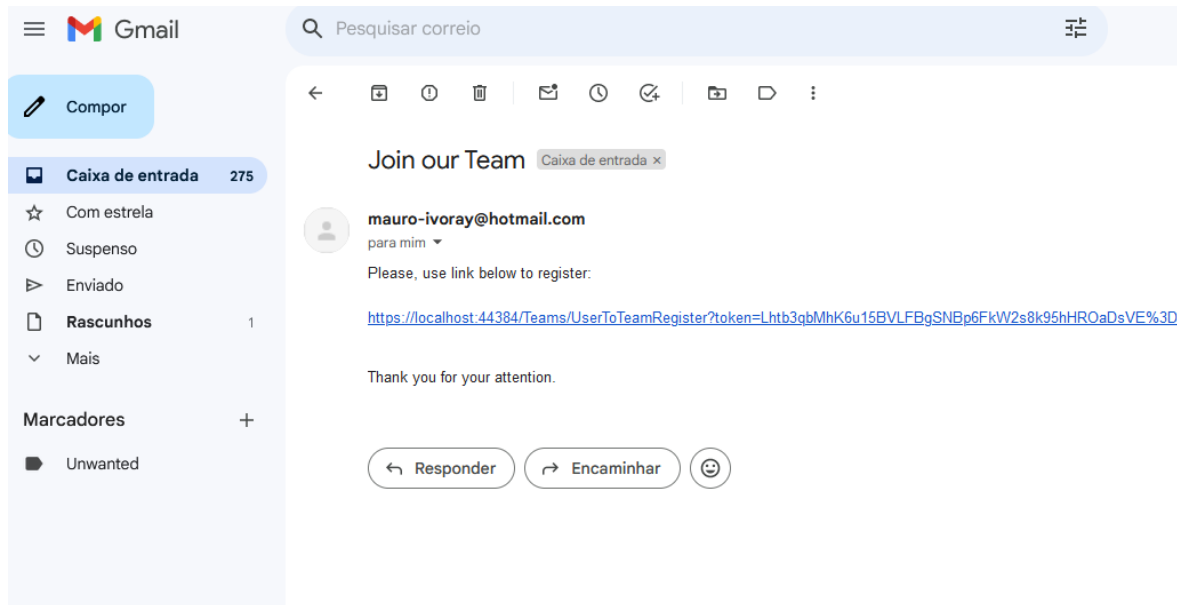


Figure 27. Member email received from team leader's invite

Figure 28 depicts the outcome of the implementation in the member management section of a team leader. It is evident that CRUD operations can be executed. Additionally, the list of added members provides interactive visibility of each team's affiliation. It is important to note that when a team leader adds a member, the member is not immediately assigned to a team. Consequently, there will be a member in the list labeled as "No team associated" until their placement is determined. In Figure 29, there is a straightforward explanation of how the pop-up is implemented when the team leader wishes to remove a certain member from the list by utilizing the delete button.

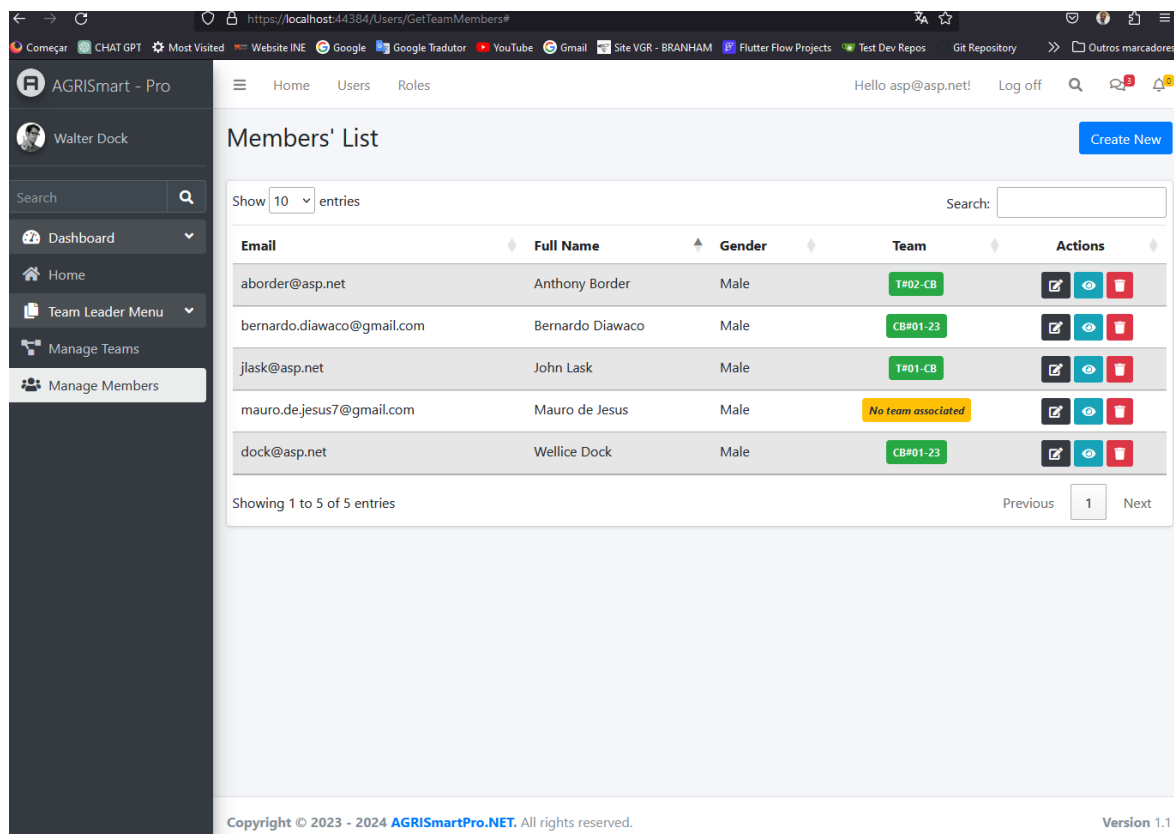


Figure 28. Admin Panel | team leader panel – members management

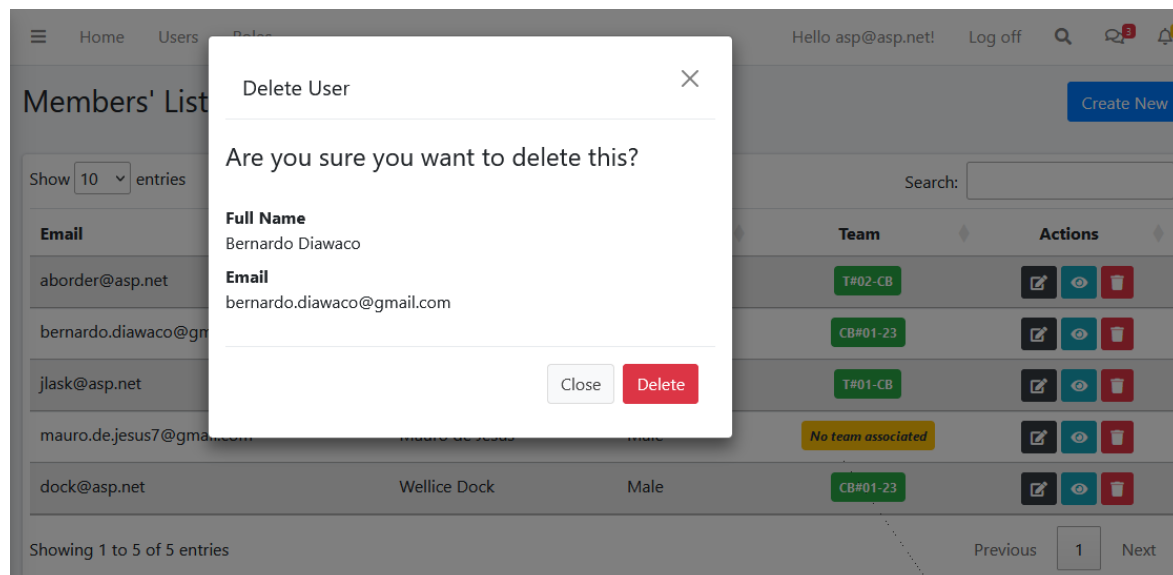


Figure 29. Admin Panel | team leader panel – members management – delete action

Figure 30 displays the outcome of applying the Disease Reports management panel specifically designed for the Team Leader. The presentation provides a comprehensive list of disease reports, which includes detailed information directly accessible from the list. It is important to mention that CRUD processes are applicable in different scenarios, enabling users to do actions such as creating, searching, deleting, editing, and viewing the information of a report. As described in the use cases, the Team Leader can view the status of their reports, including "Private" and "Public", in the list. In addition, it offers two choices: "GPS" allows users to view a map displaying the geographic place where illness reports were generated, while "Charts" provides pertinent information about the disease reports in graphical form. Lastly, there is a vital feature that pertains to the ability to acquire a table file in formats such as pdf, excel, and CSV, as well as the capability to print the data from said table. In this scenario, Excel is essential as it enables users to vary the methods of data manipulation, while also providing the opportunity to utilize powerful features like Power BI for generating reports that aid in decision-making.

# # #	Code	Crop	Disease	Created At	Is Private	Actions
4	DR-02-CB2	Orange	Blackleg and bacterial soft rot	23/11/2023	Private	[Edit] [Delete] [View]
5	DR-003-CB	Tomato	Bacterial stem rot and fruit rot	23/10/2023	Public	[Edit] [Delete] [View]
6	DR-04_CB	Tomato	Pith necrosis	23/12/2023	Public	[Edit] [Delete] [View]
7	DR-05-CB	Tomato	Syringae leaf spot	23/08/2023	Private	[Edit] [Delete] [View]
9	DR-06_CB	Tomato	Bacterial speck	23/12/2023	Private	[Edit] [Delete] [View]
10	DR-03-CB1	Potato	Blackleg and bacterial soft rot	03/01/2024	Private	[Edit] [Delete] [View]
14	DR-02_CB3	Tomato	Syringae leaf spot	05/01/2024	Public	[Edit] [Delete] [View]
15	DR-02_CB5	Tomato	Syringae leaf spot	05/01/2024	Public	[Edit] [Delete] [View]
16	DR-02_CB0	Tomato	Syringae leaf spot	05/01/2024	Public	[Edit] [Delete] [View]
41	DR03-24-CB	Orange	Bacterial spot	14/01/2024	Public	[Edit] [Delete] [View]

Figure 30. Admin Panel | team leader panel – disease reports management

Figure 31 depicts the outcome of implementing the geolocation component overseen by the Team Leader. The figure displays the precise locations where each report was generated. The map displays the reports generated by the Team Leader, reports created by team members, and all publicly available reports from other platform users. The concept of public reports entails that a different user possesses information regarding the reports and the potential to conduct diverse studies in the field.

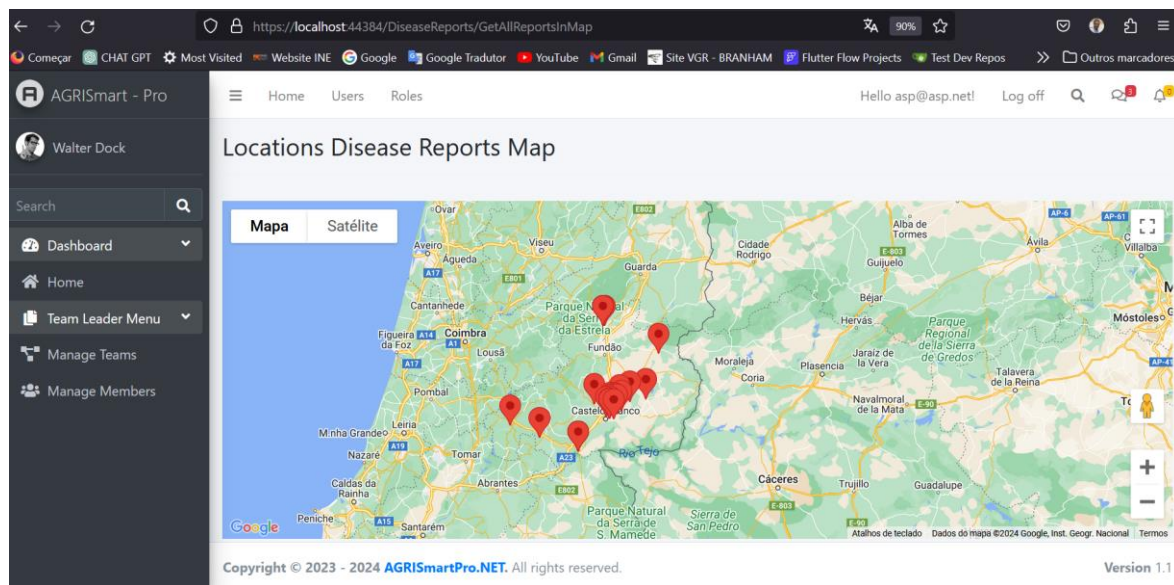


Figure 31. Admin Panel | team leader panel – disease reports management – geo-location map

As expected, Figure 32 includes interactive graphical elements while still being overseen by the Team Leader. These graphs depict the sickness reports to collect pertinent data about them. For instance, there is data that shows the "Crop with the highest number of mentions in Disease Reports," emphasizing the growing importance of the crop in field operations. Furthermore, there is data available on the "Disease with the highest number of mentions in Disease Reports," which facilitates the recognition of the most commonly seen diseases while analyzing specific agricultural crop diseases.

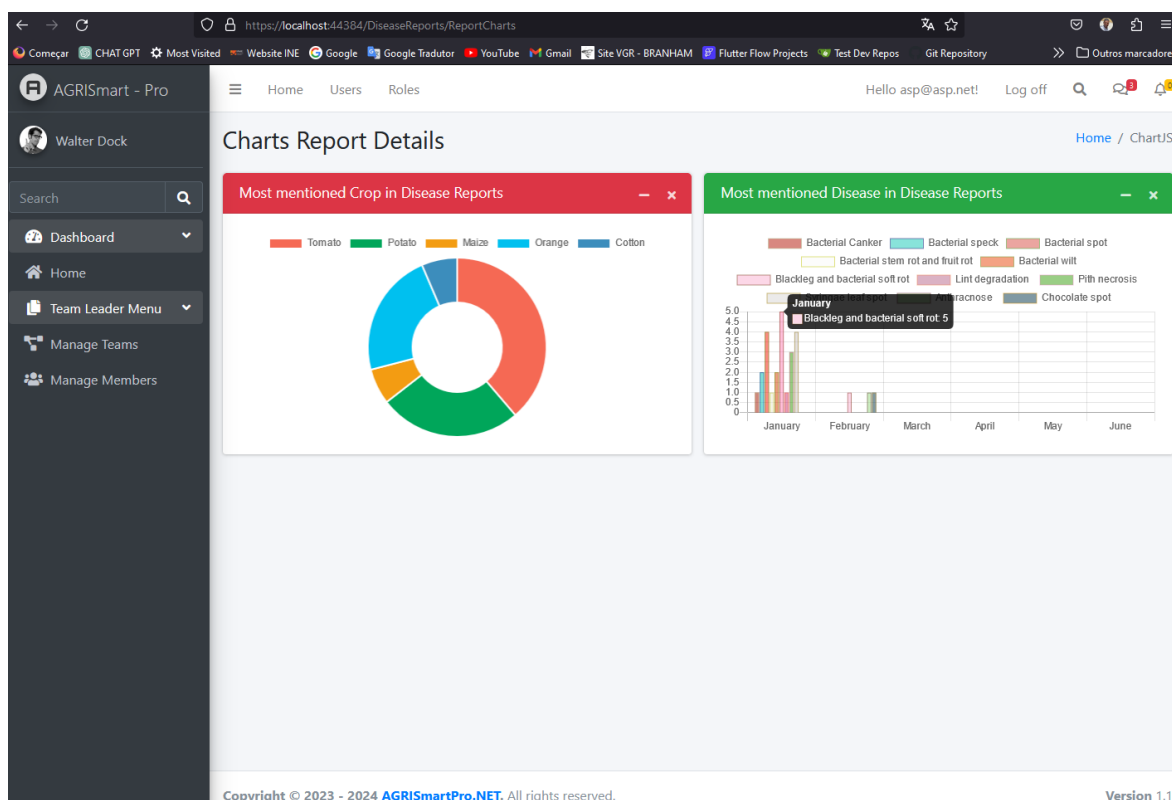


Figure 32. Admin Panel | team leader panel – disease reports management – charts

The supplementary functionality available on the main administrative page, including Pests, Anomalies, and Suggestions, are developed using the same framework and conform to the exact requirements set for illness reporting features. This guarantees a consistent approach across different modules inside the system.

Within the Admin Panel environment, Team Members have access to a range of options that closely resemble those available to the Team Leader. However, these options are subject to limitations customized to their responsibilities. Team members have limited abilities and lack the authority to modify reports. However, their role is restricted to reviewing details, which indicates their narrower operational scope in comparison to the Team Leader.

Both Team Leaders and Team Members have limited access capabilities that only allow them to view their own reports and any reports that have been made publicly accessible by other users on the site. This demarcation guarantees that data remains protected while yet being available when needed.

At the Admin Panel level, Team Members have access to features that enable them to create new reports, retrieve vital information, and evaluate data. Users can engage with an integrated map that accurately identifies geographic places, thereby improving their comprehension of spatial data distribution. In addition, users can evaluate the overall equilibrium of reports using dynamic charts that offer visual perspectives on the data. This complete toolset enhances the efficiency of management and analysis, providing support for their functions inside the organizational framework.

4.3.2 Mobile App

This section presents the specifics of the implementation of the mobile application, encompassing the selection of technology and the explanation of the development procedures. The mobile application was created utilizing the Flutter framework and the Dart language, selected for its capacity to produce top-notch applications for both Android and iOS platforms using a single source code.

Hence, the purpose of this section is to exhibit the outcome that was discussed in the use cases and storyboards regarding the desired functionalities of the mobile app. Additionally, it aims to showcase the tools used for the app's development in relation to the achieved result. Let's focus on the important aspect of autonomous disease identification in agricultural crops, which was accomplished using a machine learning algorithm. The system was trained using the Teachable Machine platform. This text will provide an analysis of the used tools for development and, most importantly, the outcome of the mobile application.

Figure 33 illustrates the result of the execution of the mobile application. It should be emphasized that mobile devices are not accessible to field workers. Agricultural field workers have the responsibility of documenting diseases, pests, and anomalies that occur in crops. The figure below depicts the app's access process, including the login screen with an unsuccessful attempt using incorrect credentials, followed by the login screen with valid credentials and a successful login message. Additionally, it displays the Team information of the field operator, specifically the Team Member.

Also, it should be emphasized that the operators who will use the mobile application will be assigned to their respective teams beforehand, as described earlier in the use case description. Operators can access the mobile app using two different methods. One option is for the team leader to register, while the other option, as previously stated during the implementation, involves the operator receiving a link to join the work team. Subsequently, this operator/team member can use the application through a login process without pre-registering on the mobile app, as they are already registered within the system.

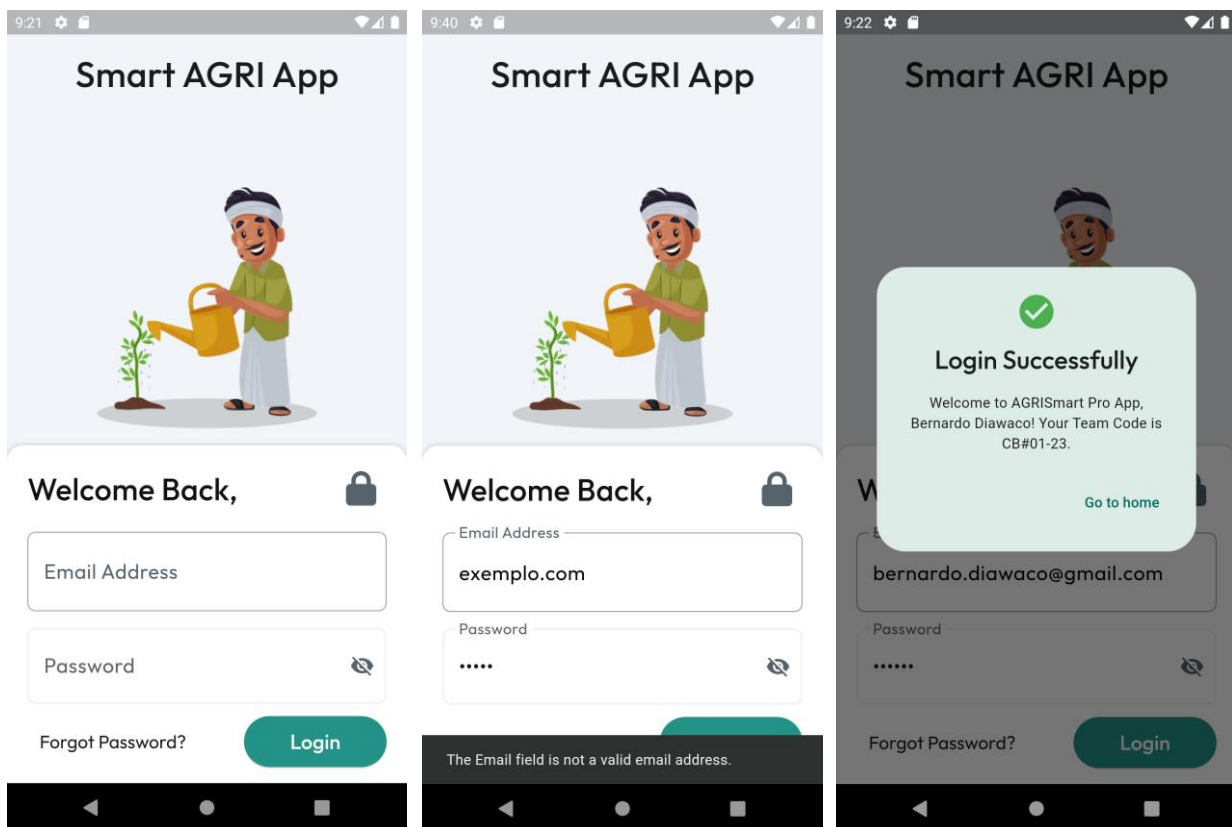


Figure 33. App mobile – Login screen

Figure 34 displays the outcome of the practical execution performed on the primary screen of the mobile application, specifically following a successful login. The image comprises four displays that present specific information related to the main page of the application, as previously outlined in the use case description. Through this interface, the user can initiate an activity by generating a report specifically for a disease, pest, or abnormality. The user has the ability to obtain history of disease and pest reports that were made in the past. Additionally, by selecting the "See details" option, the user can access more comprehensive information pertaining to the disease report, including a photograph and the geographical location. Furthermore, a navigation bar is included below to facilitate the user's access to a section dedicated to providing more comprehensive details about the data in their reports, specifically pertaining to geolocation.

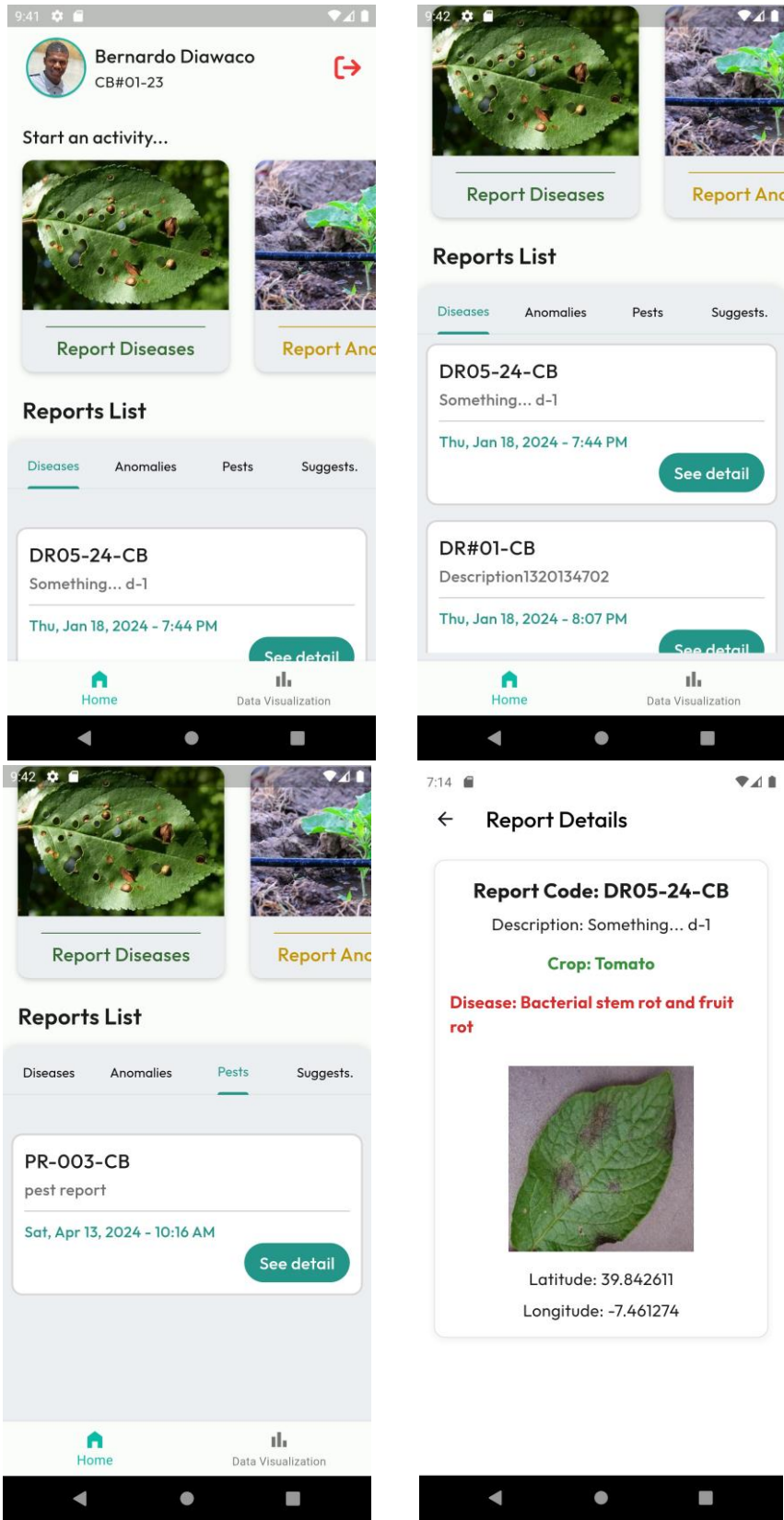


Figure 34. App mobile – Main screen and disease report details

Figure 35 displays the implemented outcome for the options available to create a Disease Report. The figure consists of 4 screens and is visible on the main screen. By selecting the Disease Report option on the main screen, users can view the options available for creating a report, as shown in the image. It is important to mention that this screen allows users to view the available Crops for automatic disease identification. Subsequently, it is possible to select from the available Crops and then decide whether you wish to capture a shot or opt for a pre-existing photo from the collection. The sample provided below demonstrates a photograph extracted from the gallery. Upon selecting the application, it promptly detects the ailment and displays the precision of the outcome. Once the condition has been discovered, you can use the "Check Details" button to access the internet and find further information about the identified aid, if needed to enhance the report.

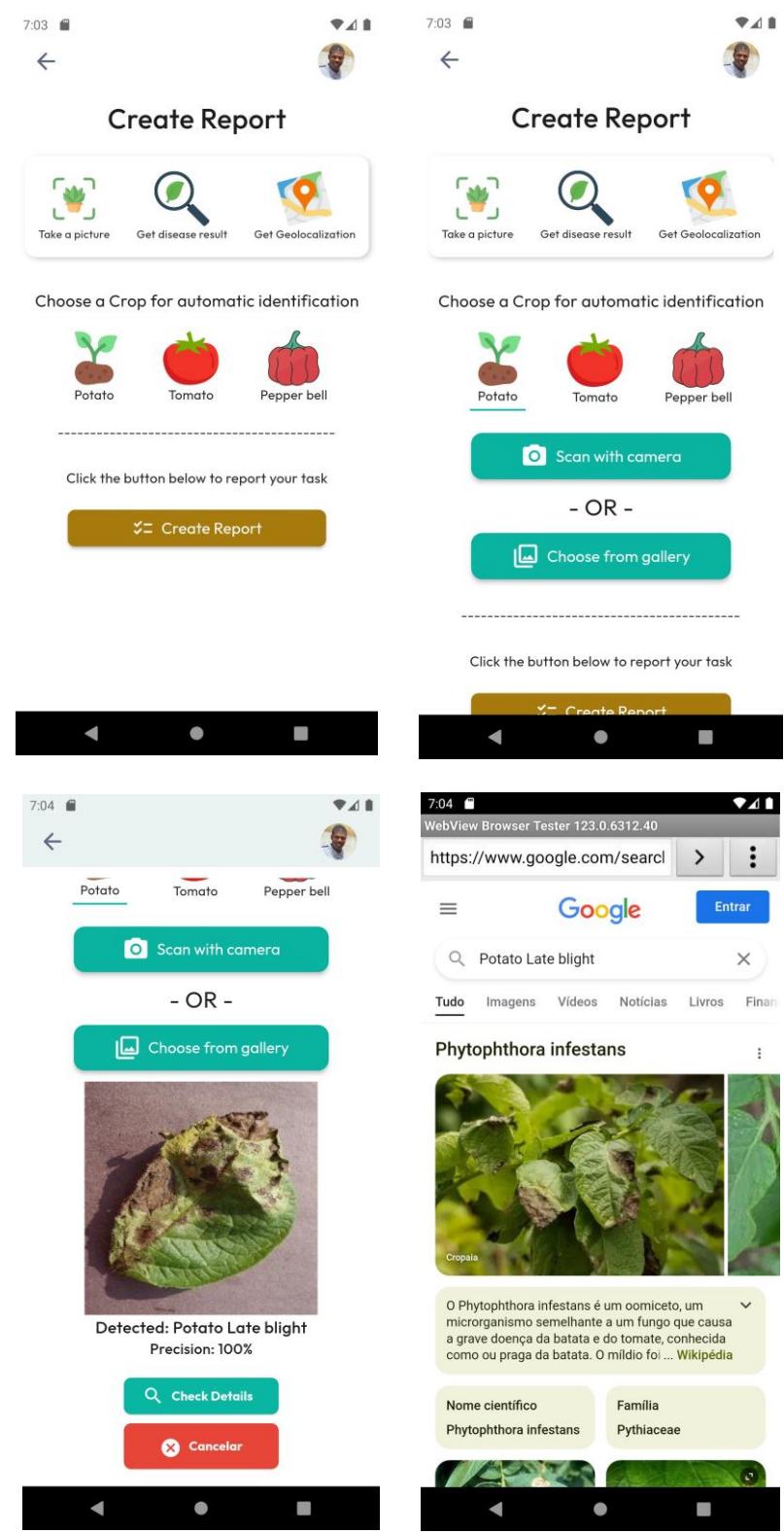


Figure 35. App mobile – Disease report menu-option screens

Figure 36 depicts the outcome of the disease report creation implementation. This image, composed of four screens, displays the pallets provided for the generation of a report. The first and second screens display the essential fields for the report, such as the selection of a crop and a disease, among others. The third and fourth panels display the choice to either upload an image or use the camera to include a photo in the report. The user is presented with a pre-selected image that can be removed if necessary. They are given the opportunity to choose whether the report will be public or private. Finally, there is a "Create" button that allows the user to upload the data to the server.

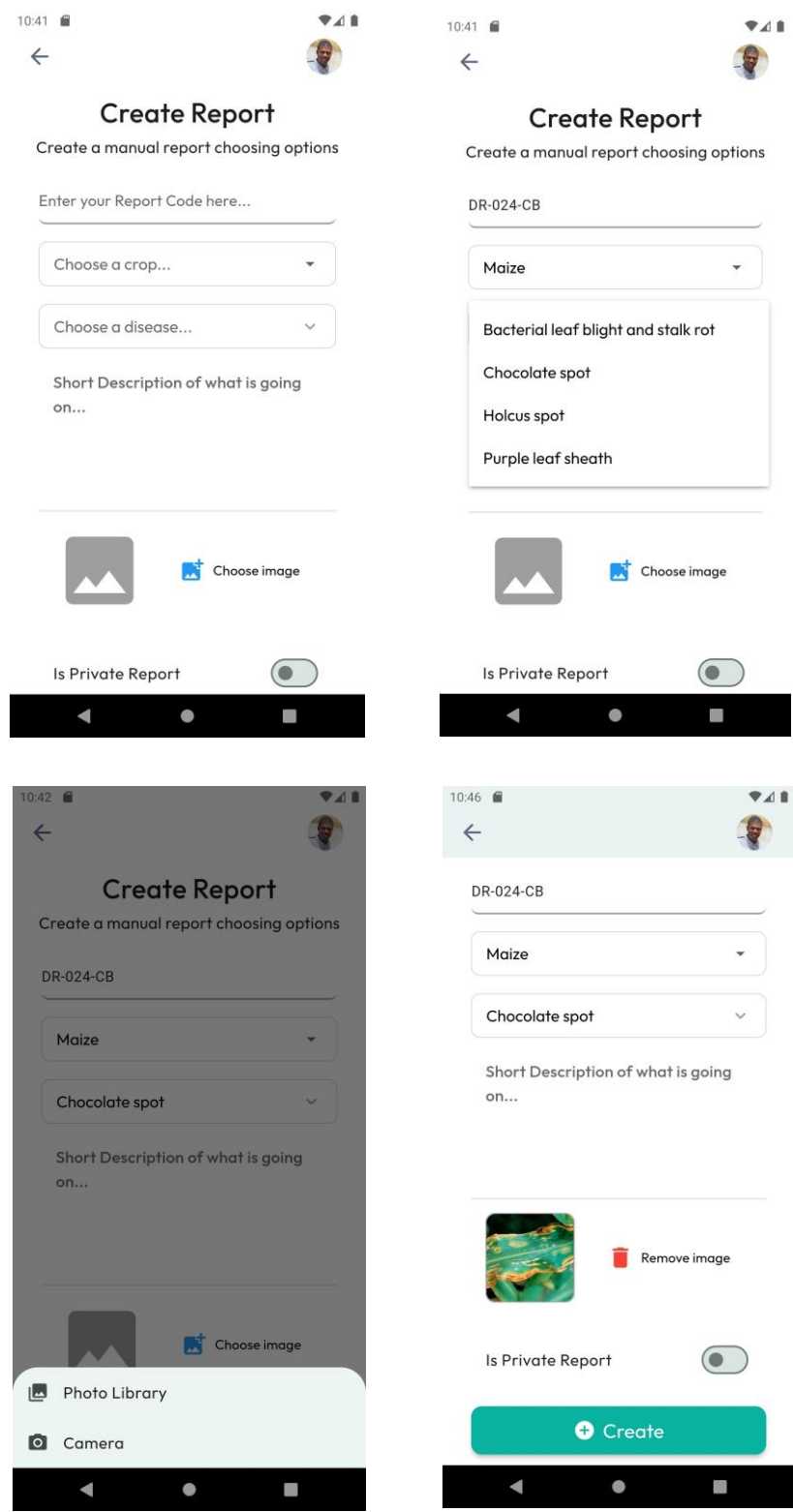


Figure 36. App mobile – Disease report creation menu-option screens

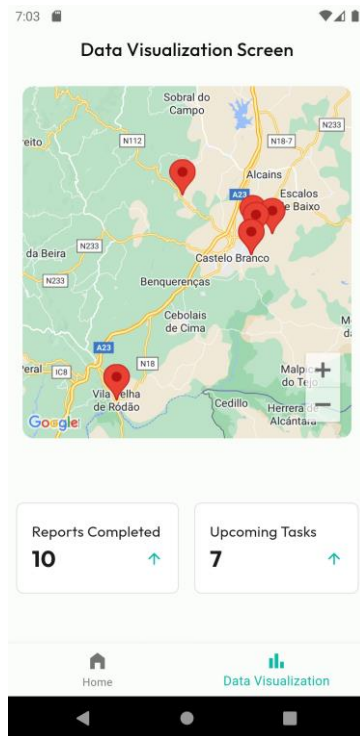


Figure 37 depicts the execution of the data visualization component of the mobile application, notably focusing on the geographical location of reports generated on the platform. This screen provides the user with access to the geolocation of their reports, allowing them to determine the specific locations where they worked and where each report is categorized. Additionally, it is feasible to access information on the quantity of reports conducted and the number of jobs scheduled for completion on a given day.

Figure 37. App mobile – Data Visualization navbar screen

Subsequently, the machine learning model employed for the automated detection of diseases in this system, particularly in the mobile application, will be introduced. The Google Teachable Machine platform was used to construct a model that automates the diagnosis of diseases using leaves, as previously stated in the discussion of the techniques employed in implementing this app.

Figure 38 displays the Teachable Machine interface, an online tool created by Google that enables the development of machine learning models in a straightforward and user-friendly manner, without requiring expertise in programming or data science. This figure displays the Teachable Machine interface, which is partitioned into multiple sections.

Image Classes

- The dataset consists of 178 picture samples of healthy tomato leaves.
- The dataset consists of 287 picture samples of tomato leaves exhibiting the illness known as late blight.
- Potato Healthy: Comprises 152 image samples depicting the leaves of healthy potatoes.
- The dataset "Potato Late blight" has 187 picture samples of potato leaves exhibiting the same illness.

In addition to the photographs shown in Figure 38, there are other images categories not included. Specifically, there is a category of images including Crop Pepper Bell, which was also considered when developing the model on the platform.

In the platform, each class offers the possibility to include additional photographs either through a camera or by uploading them, which helps in gathering data for training the model.

Training section

- Train Model: Press the button to initiate the training process using the specified photos.
- Additional features:
 - Number of epochs: Set to 50. Epochs indicate the total number of times the entire dataset is iterated during the training process. Augmenting the number of epochs can enhance the model's accuracy, but it can also result in overfitting.
 - Batch Size: Configured as 16. The batch size refers to the number of samples that the model processes before updating its parameters. Reducing the batch size can enhance training stability, but it may result in longer training times.
 - Learning rate: The learning rate is set to 0.001. The learning rate determines the extent to which the model's weights are modified according to the gradient of the loss function throughout the training process. An excessively high learning rate can lead to the model

converging too rapidly to a suboptimal solution, whilst an excessively low rate can result in delayed training.

Preview and export section

Once the model has been trained, is possible to observe it in the Preview area. The export model feature allows users to save the trained model, with a particular focus on the TFLite format. TFLite is a variant of TensorFlow that is specifically designed for optimal performance on mobile and embedded devices.

Model Export

Once the training is deemed satisfactory, the model can be exported. The TFLite format is particularly advantageous for deploying the model on mobile devices or embedded systems, enabling plant disease detection applications to be executed immediately in the field, without requiring a continuous internet connection. This function enables the utilization of the mobile app without an internet connection.

The primary algorithms utilized by Teachable Machine

Convolutional Neural Networks (CNNs): Are crucial for image processing as they excel at identifying and analyzing patterns and features within submitted photographs.

Transfer Learning: Teachable Machine frequently employs pre-trained models and refines them using user-provided data, resulting in time and computing resource savings.

Teachable Machine streamlines the process of developing machine learning models, enabling educators, students, and professionals to adopt AI solutions even without extensive technical expertise. The user-friendly interface, along with the sophisticated configuration options, enables you to swiftly and effectively generate personalized templates for a wide range of applications.

Figure 39 illustrates the conclusion of the model training process. The image displays all the pre-trained image classes for the specified crops in the system, allowing

for testing before exporting the model. Figure 40 displays the different options available for exporting the model. In this scenario, the chosen option was used, which is commonly used for testing on the Android platform. This method involves extracting a quantized model and subsequently invoking the assets in the mobile application to ensure optimal functionality.

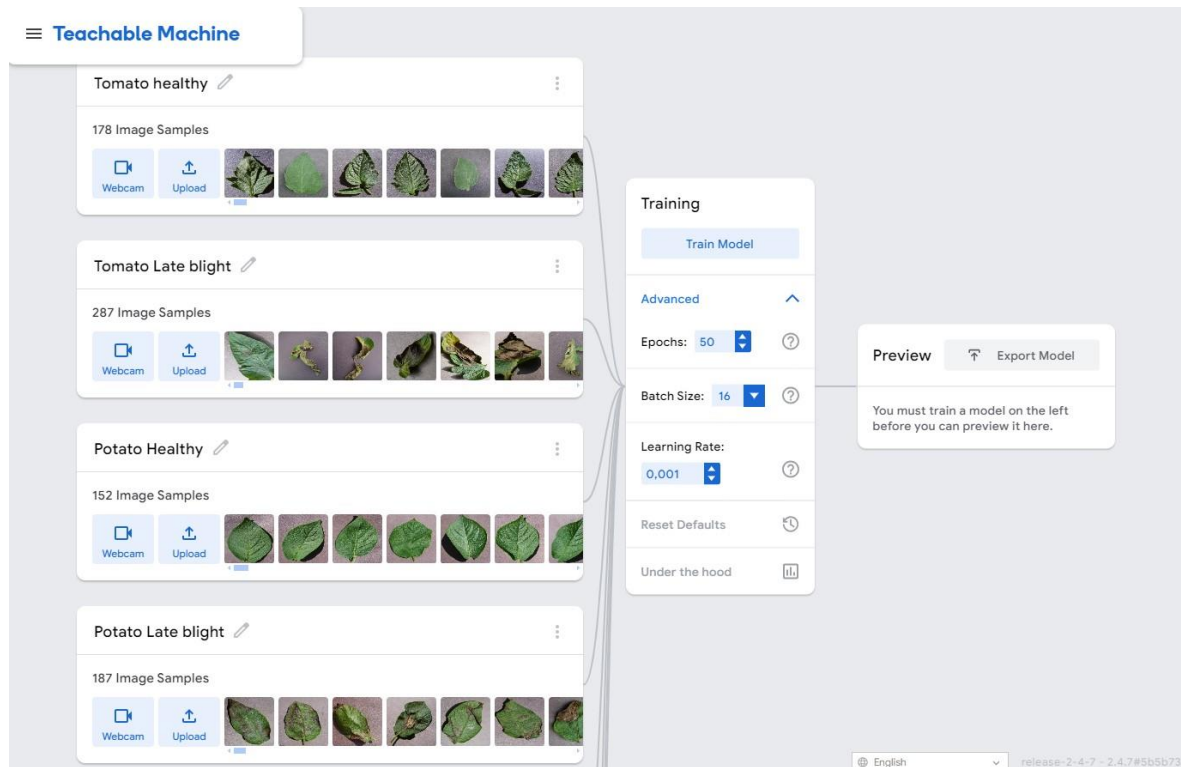


Figure 38. Teachable Machine Screen – images class creation

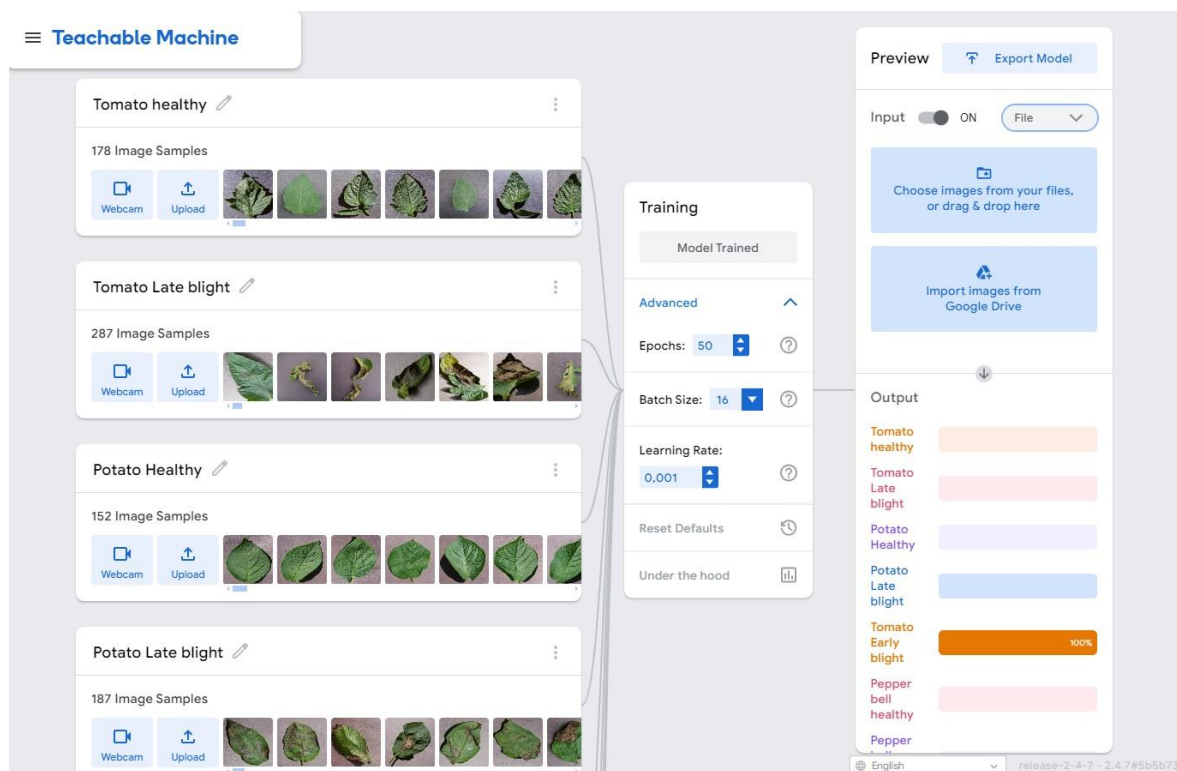


Figure 39. Teachable Machine Screen – model training completed

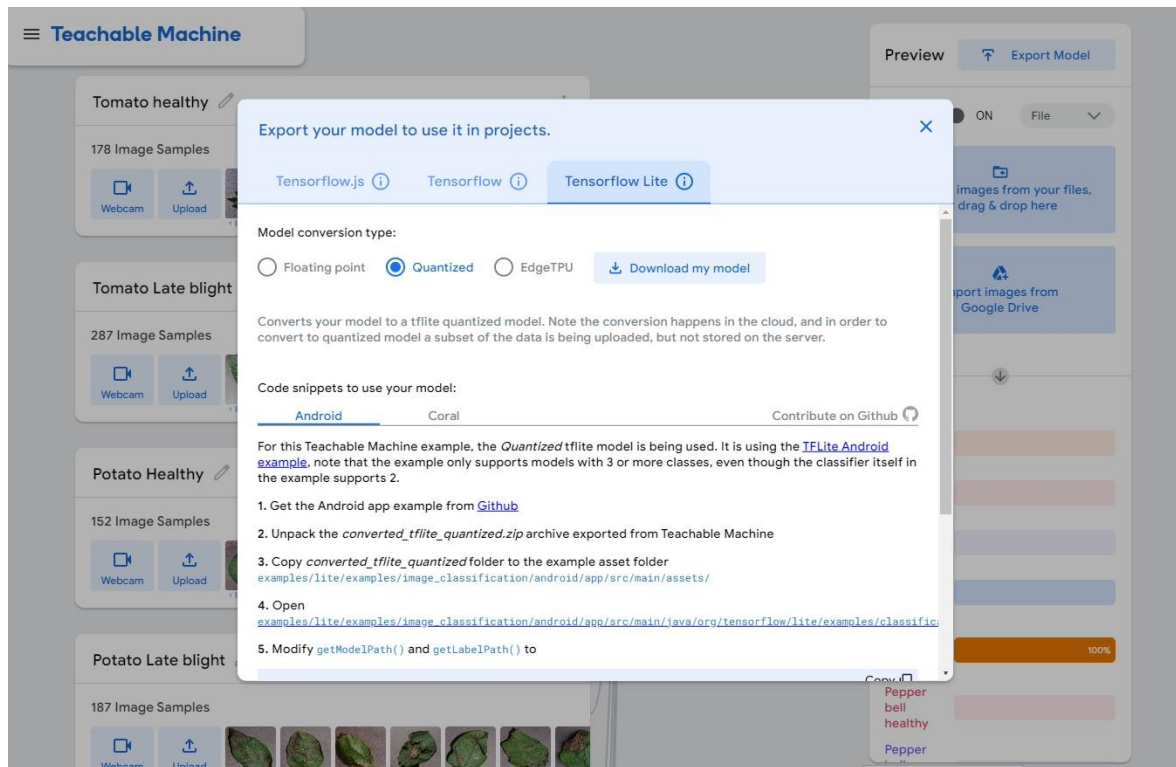


Figure 40. Teachable Machine Screen – model export

4.4 Tests and Results

Using a set of photos distinct from those used in training, tests were conducted to assess the efficacy of the plant disease detection model produced with the Teachable Machine tool. To guarantee the model could generalize effectively to hitherto unheard-of data, the test photos were selected at random from the internet.

The proposed model showed accuracy in identifying diseases in plant leaves, according to the experiments conducted. These are the procedures and findings of the testing process.

Model Configuration:

- Images of both healthy and diseased tomato and potato leaves were utilized, as Figures 38, and 39 illustrate.
- Training parameters for the model included a learning rate of 0.001, batch size of 16 and 50 epochs (Figure 38).

Model Export and Implementation:

- Following instruction, the model was exported in TensorFlow Lite quantized form to be included into the mobile app (Figure 40).

Test results:

- With 100% accuracy in representative samples, the Mobile App was able to properly identify the disease on plant leaves. Figure 35 shows very high accuracy detection of a particular disease—late blight in potatoes.
- Using photos not included in the original dataset, further tests were conducted to validate the model by accurately spotting diseases in the additional photographs.

The results indicate the created model has considerable generalizing ability, so proving efficiency on both seen and unseen data. The system's strengths are validated by the continuous accuracy in several tests, which distinguishes itself as a potential instrument for automated plant disease identification.

In addition, the geolocation feature was tested on both an Android emulator and a real mobile phone. This approach was implemented to guarantee the accurate functioning of the geolocation system across different scenarios and hardware setups.

In order to evaluate geolocation on the Android emulator, it was imperative to set up the emulator's GPS. The configuration procedure comprises the subsequent steps:

- Launch the Android Virtual Device (AVD) Manager: This software enables users to generate and customize emulators.
- Choose a device: Select the emulated device on which the test will be executed.
- To specify the location, navigate to the AVD control panel and select the option to manually input GPS coordinates. Users input desired coordinates to replicate various places.

Furthermore, geolocation was assessed on an actual mobile device to determine the precise correctness of the recorded GPS coordinates, in addition to testing on the emulator. This entailed the installation of the application on the device and conducting a simulation that simulated an operator in the field. During the simulation, the accuracy of the coordinates recorded by the app was compared to the GPS coordinates acquired from high-precision devices.

Geolocation test results:

The geolocation feature of the tests revealed satisfactory accuracy. Figure 37 shows the GPS values noted during the tests. While on the mobile device the GPS depends on the device itself, these points were manually recorded in the emulator (whereas in the system may precisely capture and present the location of diseases, pests and anomalies discovered on the plantations).

- Accuracy in the Emulator: As demonstrated in Figure 37, the manually entered coordinates in the emulator showed on the application map exactly.
- Accuracy in the Physical Device: Generally less than five meters, the observed margin of error is reasonable for uses in agriculture.

Geolocation testing essentially showed that, in both real-world and simulated contexts, the functionality is accurate. This guarantees that location data will enable next users of the app to make wise choices on crop management. Geolocation's involvement as a

tool in the system helps the application to be generally more effective by enabling quick and accurate actions.

4.5 Final Remarks

This chapter presents the architecture, the development tools, and implementation techniques used in the system. Because of the modularity and scalability of the system architecture, component communication was made effective and maintainable. At the heart of this design is a RESTful web service locally hosted on a server that allows smooth connections to some client interfaces, including web browsers and mobile apps. With updates enabled without interfering with the user experience, this design encourages flexibility.

Furthermore, covered in the chapter were the development tools. Because of how well it can build dynamic, data-driven websites, the ASP.NET MVC framework was selected for the web-based Admin Panel. Because it allows the development of cross-platform apps, Flutter with Dart was chosen for the development of the mobile application. To improve the system's precision at identifying diseases and pests, machine learning models were also included.

Regarding the application, the chapter gave a summary of the process by which the theoretical designs were converted into useful, functional parts. It included methods and best practices as it went over the creation of the Mobile App and the Admin Panel. To guarantee system accuracy and dependability, this involved incorporating machine learning models, coding standards, and setting up development environments.

Including a thorough testing period has also shown the Teachable Machine model's accuracy and dependability. The tests, which used a different collection of photos not included in the training set, revealed 100% accuracy in disease diagnosis, therefore confirming the generalizing capacity of the model. This significant level of performance emphasizes the system's possibilities as a dependable instrument for automated plant disease identification, therefore benefiting farmers and agricultural managers in terms of timely intervention and crop management.

Moreover extensively examined in the Android emulator and a real mobile device was the geolocation capability. With a margin of error usually less than five meters,

which is appropriate for agricultural uses, the tests proved great accuracy in recording and presenting GPS data. The effective implementation of geolocation guarantees that the application can precisely track and visualize the locations of found diseases, pests, and anomalies, so improving the general management and intervention strategies for crop health as shown in Figure 37.

This chapter generally did an outstanding job of illustrating the technical labor and meticulous planning needed to produce a dependable and user-friendly system ready for practical use. It also deftly caught the road from idea to reality. Geolocation's presence as a fundamental tool emphasizes even more the system's all-encompassing character as it gives users accurate, practical information to raise agricultural sustainability and productivity.

5. Conclusions

The creation of a complete system for disease, pest, and anomaly identification and geolocation in agricultural crops was the main goal of this dissertation. Using cutting edge technology like machine learning, the system seeks to improve the effectiveness and precision of disease diagnosis and management in agriculture.

The automatic identification of diseases from photos of plant leaves, their record and geolocation, and the visualization and analysis of the gathered data were the defined goals that directed the project. Flexibility and user-specific capabilities were ensured by the system's design to be user profile-based.

An assessment of the literature and technology now used in automated disease detection in agriculture started the investigation. The important progress achieved in the identification of plant diseases using machine learning was emphasized in this review. It also made clear how urgently systems able to combine these technologies are needed to offer precise, scalable, and real-time agricultural management solutions.

Both functional and non-functional requirements had to be carefully examined throughout the requirements and modeling stage. Non-functional requirements described the operational criteria including performance, dependability, and usability, whereas functional requirements specified the precise actions and capabilities the system had. Using storyboards and use case diagrams, these functional requirements were visualized to improve stakeholder understanding and communication.

Scalability and modularity dominated the system architecture design process. The system's foundation was a RESTful web API that made it possible to integrate with mobile apps and web browsers among other client interfaces with ease. The end user experience is not interrupted by flexible and independent business logic updates supported by this design.

Because of its many capabilities for creating dynamic, data-driven websites, the ASP.NET MVC framework was chosen for the web-based Admin Panel. Because Flutter with Dart is so effective at building cross-platform apps, it was used to produce mobile apps. Integrated machine learning models increased the system's ability to precisely identify diseases and pests, therefore raising its overall efficacy.

The implementation phase turned theoretical designs into real-world applications, and a thorough, step-by-step process made sure the system satisfied all necessary specifications. To guarantee accuracy and dependability, this stage included integrating machine learning models, establishing development environments, and following coding guidelines.

Because it automatically detects diseases, pests, and anomalies, the system provides a complete tool for controlling agricultural health. It gives farmers and other agricultural managers strong tools to track crop health, see patterns in the data, and make wise choices that eventually increase agricultural productivity and sustainability.

The system has certain limits even with the success attained. One of the primary limitations at the Mobile App platform was that the automatic disease identification or detection depends on the user capturing a picture of the afflicted plant to view the outcome. The intended implementation was that the system could also operate as a scanner, in which case it would not only be required to capture a picture but also direct the camera at the plant and instantly observe the disease or accuracy. Technical constraints and resources during prototype development prevented this capability. Moreover, the absence of thorough databases for other agricultural crops restricted the app's ability to identify three kinds of crops since these were the only ones accessible in the used datasets. Another major constraint was the absence of datasets tailored specifically for pests, therefore limiting the system's ability to detect more diverse agricultural issues.

Furthermore, discovered in the Web system were some limitations, such the ability of dashboards with more exact information visualization of data from field-made reports to avoid using other data management tools.

For future work, several improvements can be made to overcome current limitations. First, the application of sophisticated computer vision and image processing methods can enable real-time disease detection, thereby turning the camera of the device into a scanner capable of spotting issues without the need of obtaining a stationary picture.

Furthermore, crucial for raising the accuracy and scope of the application is the extension and enrichment of datasets for pests as well as for crops. This will enable the system to identify several diseases and pests in many different agricultural crops.

Let the application operate offline as another major enhancement would be possible. Farmers operating in places with restricted internet access would especially benefit from this so they may fully use the system without interruption.

The system can also be enlarged to incorporate fresh capabilities including tailored therapy suggestions derived from data collection. Ensuring the system efficiently satisfies the needs of the target audience depends on improving the user interface and doing thorough testing involving more end users.

Last but not least, interaction with other developing technologies like IoT sensors and drones can provide an even more complete and thorough picture of the state of agricultural crops, therefore supporting even more exacting precision agriculture and sustainability.

Finally, this dissertation has resulted in a system that combines technologies to tackle important issues in agricultural management. Technology offers a strong way to recognize and control crop diseases, pests, and anomalies by using machine learning, image processing, and geolocation capabilities. In addition to furthering agricultural technology, this initiative has the potential to have a big influence on food security and agricultural sustainability.

References

- [1] M. Altieri, *Agroecology: The Science Of Sustainable Agriculture*, 2nd Edition. 2018.
- [2] Ahmed Abdelmoamen Ahmed, Suhib Al Omari, R. Awal, A. Fares, and M. Chouikha, "A Distributed System For Supporting Smart Irrigation Using IoT Technology," *ProQuest*, Jul. 2021.
- [3] Jun Sun; Yu Yang; Xiaofei He; Xiaohong Wu, "Northern Maize Leaf Blight Detection Under Complex Field Environment Based on Deep Learning," *IEEE Explorer*, Feb. 2020.
- [4] Aditya Sinha; Rajveer Singh Shekhawat, "Review of Image Processing Approaches for Detecting Plant Diseases," *IET - The Institution of Engineering and Technology*, May 2020.
- [5] Jinya Su;, Dewei Yi;, Baofeng Su;, Zhiwen Mi;, Cunjia Liu;, and Xiaoping Hu, "Aerial Visual Perception in Smart Farming: Field Study of Wheat Yellow Rust Monitoring," *IEEE Explorer*, Mar. 2020.
- [6] N. Anderson, "What Is the Waterfall Methodology?," Built In. Accessed: Jun. 07, 2024. [Online]. Available: <https://builtin.com/software-engineering-perspectives/waterfall-methodology>
- [7] M. Francisco, F. Ribeiro, J. Metrôlho, and R. Dionisio, "Algorithms and Models for Automatic Detection and Classification of Diseases and Pests in Agricultural Crops: A Systematic Review," *Aplied Sciences*, Mar. 2023, Accessed: Jun. 04, 2024. [Online]. Available: <https://www.mdpi.com/2076-3417/13/8/4720>
- [8] M. K. N. and M. M. M. Karthick, D. Vijayalakshmi, "Machine Learning Approaches for Automatic Disease Detection from Paddy Crops - A Review," *International Journal of Engineering Trends and Technology*, vol. 70, no. 12, pp. 392–405, 2022, doi: 10.14445/22315381/IJETT-V70I12P237.
- [9] T. Sharma, P. Kaur, J. Chahal, and H. Sharma, "Classification of Rice Leaf Diseases based on the Deep Convolutional Neural Network Architectures: Review," *AIP Conf Proc*, vol. 2451, no. October, 2022, doi: 10.1063/5.0095670.
- [10] T. Domingues, T. Brandão, and J. C. Ferreira, "Machine Learning for Detection and Prediction of Crop Diseases and Pests: A Comprehensive Survey," *Agriculture*, vol. 12, no. 9, 2022, doi: 10.3390/agriculture12091350.
- [11] R. Manavalan, "Towards an intelligent approaches for cotton diseases detection: A review," *Comput Electron Agric*, vol. 200, p. 107255, 2022, doi: <https://doi.org/10.1016/j.compag.2022.107255>.

- [12] B. Tugrul, E. Elfatimi, and R. Eryigit, "Convolutional Neural Networks in Detection of Plant Leaf Diseases: A Review," *Agriculture*, vol. 12, no. 8, 2022, doi: 10.3390/agriculture12081192.
- [13] R. Thangaraj, S. Anandamurugan, P. Pandiyan, and V. K. Kaliappan, "Artificial intelligence in tomato leaf disease detection: a comprehensive review and discussion," *Journal of Plant Diseases and Protection*, vol. 129, no. 3, pp. 469–488, 2022, doi: 10.1007/s41348-021-00500-8.
- [14] D. Bhise, S. Kumar, and H. Mohapatra, "Review on deep learning-based plant disease detection," in *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, 2022, pp. 1106–1111. doi: 10.1109/ICECA55336.2022.10009290.
- [15] W. Setiawan, E. M. S. Rochman, B. D. Satoto, and A. Rachmad, "Machine Learning and Deep Learning for Maize Leaf Disease Classification: A Review," *J Phys Conf Ser*, vol. 2406, no. 1, p. 12019, 2022, doi: 10.1088/1742-6596/2406/1/012019.
- [16] N. T. Sinshaw, B. G. Assefa, S. K. Mohapatra, and A. M. Beyene, "Applications of Computer Vision on Automatic Potato Plant Disease Detection: A Systematic Literature Review," *Comput Intell Neurosci*, vol. 2022, p. 7186687, 2022, doi: 10.1155/2022/7186687.
- [17] V. Sharma, A. K. Tripathi, and H. Mittal, "Technological Advancements in Automated Crop Pest and Disease Detection: A Review & Ongoing Research," in *2022 International Conference on Computing, Communication, Security and Intelligent Systems (IC3SIS)*, 2022, pp. 1–6. doi: 10.1109/IC3SIS54991.2022.9885605.
- [18] S. Saifi, B. Mukund, D. Alam, and R. Khanam, "A Review on Plant Leaf Disease Detection using Deep Learning," in *2022 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES)*, 2022, pp. 101–107. doi: 10.1109/CISES54857.2022.9844271.
- [19] R. Kumar, A. Chug, A. P. Singh, and D. Singh, "A Systematic Analysis of Machine Learning and Deep Learning Based Approaches for Plant Leaf Disease Classification: A Review," *J Sens*, vol. 2022, p. 3287561, 2022, doi: 10.1155/2022/3287561.
- [20] V. Kumar and V. Laxmi, "Pests Detection Using Artificial Neural Network and Image Processing: A Review," in *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, 2022, pp. 462–467. doi: 10.1109/ICSCDS53736.2022.9760976.
- [21] L. Li, S. Zhang, and B. Wang, "Plant Disease Detection and Classification by Deep Learning - A Review," *IEEE Access*, vol. 9, pp. 56683 – 56698, 2021, doi: 10.1109/ACCESS.2021.3069646.

- [22] R. N. Jogekar and N. Tiwari, "A Review of Deep Learning Techniques for Identification and Diagnosis of Plant Leaf Disease," *Smart Innovation, Systems and Technologies*, vol. 182, pp. 435 – 441, 2021, doi: 10.1007/978-981-15-5224-3_43.
- [23] L. C. Ngugi, M. Abelwahab, and M. Abo-Zahhad, "Recent advances in image processing techniques for automated leaf pest and disease recognition – A review," *Information Processing in Agriculture*, vol. 8, no. 1, pp. 27–51, 2021, doi: <https://doi.org/10.1016/j.inpa.2020.04.004>.
- [24] H. F. Pardede, E. Suryawati, D. Krisnandi, R. S. Yuwana, and V. Zilvan, "Machine Learning Based Plant Diseases Detection: A Review," *2020 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)*, pp. 212–217, 2020.
- [25] R. Manavalan, "Automatic identification of diseases in grains crops through computational approaches: A review," *Comput Electron Agric*, vol. 178, p. 105802, 2020, doi: <https://doi.org/10.1016/j.compag.2020.105802>.
- [26] M. Nagaraju and P. Chawla, "Systematic review of deep learning techniques in plant disease detection," *International Journal of System Assurance Engineering and Management*, vol. 11, no. 3, pp. 547–560, 2020, doi: 10.1007/s13198-020-00972-1.
- [27] M. Cardim Ferreira Lima, M. E. de Almeida Leandro, C. Valero, L. C. Pereira Coronel, and C. O. Gonçalves Bazzo, "Automatic Detection and Monitoring of Insect Pests—A Review," *Agriculture*, vol. 10, no. 5, 2020, doi: 10.3390/agriculture10050161.
- [28] M. J. Page *et al.*, "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews," *BMJ*, p. n71, Mar. 2021, doi: 10.1136/bmj.n71.
- [29] E. Scopus, "Scopus." Accessed: Dec. 05, 2022. [Online]. Available: <https://www.scopus.com/search/form.uri?display=basic#basic>
- [30] Clarivate, "Web of Science." Accessed: Dec. 05, 2022. [Online]. Available: <https://www.webofscience.com/wos/woscc/basic-search>
- [31] M. J. Page *et al.*, "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews," *BMJ*, vol. 372, 2021, doi: 10.1136/bmj.n71.
- [32] N. Zhao, L. Zhou, T. Huang, M. F. Taha, Y. He, and Z. Qiu, "Development of an automatic pest monitoring system using a deep learning model of DPeNet," *Measurement*, vol. 203, p. 111970, 2022, doi: 10.1016/j.measurement.2022.111970.
- [33] S. Lee, G. Choi, H.-C. Park, and C. Choi, "Automatic Classification Service System for Citrus Pest Recognition Based on Deep Learning," *Sensors*, vol. 22, no. 22, p. 8911, 2022, doi: 10.3390/s22228911.

- [34] C. Chen, Y. Liang, L. Zhou, X. Tang, and M. Dai, "An automatic inspection system for pest detection in granaries using YOLOv4," *Comput Electron Agric*, vol. 201, p. 107302, 2022, doi: 10.1016/j.compag.2022.107302.
- [35] R. Yakkundimath, G. Saunshi, B. Anami, and S. Palaiah, "Classification of Rice Diseases using Convolutional Neural Network Models," *Journal of The Institution of Engineers (India): Series B*, vol. 103, no. 4, pp. 1047–1059, 2022, doi: 10.1007/s40031-021-00704-4.
- [36] S. Chakraborty, H. Kodamana, and S. Chakraborty, "Deep learning aided automatic and reliable detection of tomato begomovirus infections in plants," *J Plant Biochem Biotechnol*, vol. 31, no. 3, pp. 573–580, 2022, doi: 10.1007/s13562-021-00732-7.
- [37] E. Acar, O. F. Ertugrul, E. Aldemir, and A. Oztekin, "Automatic identification of cassava leaf diseases utilizing morphological hidden patterns and multi-feature textures with a distributed structure-based classification approach," *Journal of Plant Diseases and Protection*, vol. 129, no. 3, pp. 605–621, 2022, doi: 10.1007/s41348-022-00583-x.
- [38] Y. Zhang, S. Wa, L. Zhang, and C. Lv, "Automatic Plant Disease Detection Based on Tranvolution Detection Network With GAN Modules Using Leaf Images," *Front Plant Sci*, vol. 13, 2022, doi: 10.3389/fpls.2022.875693.
- [39] V. Sathiya, M. S. Josephine, and V. Jeyabalaraja, "An automatic classification and early disease detection technique for herbs plant," *Computers and Electrical Engineering*, vol. 100, p. 108026, 2022, doi: 10.1016/j.compeleceng.2022.108026.
- [40] R. G. Dawod and C. Dobre, "Automatic Segmentation and Classification System for Foliar Diseases in Sunflower," *Sustainability*, vol. 14, no. 18, p. 11312, 2022, doi: 10.3390/su141811312.
- [41] G. M. Brindha, K. K. Karishma, J. Nivetha, and B. Vidhya, "Automatic Detection of Citrus Fruit Diseases Using MIB Classifier," in *2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC)*, IEEE, 2022, pp. 1111–1116. doi: 10.1109/ICESC54411.2022.9885702.
- [42] H. Mustafa *et al.*, "Pepper bell leaf disease detection and classification using optimized convolutional neural network," *Multimed Tools Appl*, 2022, doi: 10.1007/s11042-022-13737-8.
- [43] A. Verma, S. Shekhar, and H. Garg, "Plant Disease Classification Using Deep Learning Framework," in *2022 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES)*, IEEE, 2022, pp. 512–518. doi: 10.1109/CISES54857.2022.9844352.

- [44] S. Qian, J. Du, J. Zhou, C. Xie, L. Jiao, and R. Li, "An effective pest detection method with automatic data augmentation strategy in the agricultural field," *Signal Image Video Process*, 2022, doi: 10.1007/s11760-022-02261-9.
- [45] P. Sharma, "Automatic Classification and Diseases Identification of Indian Medicinal Plants Using Deep Learning Models," *ECS Trans*, vol. 107, no. 1, pp. 4631–4639, 2022, doi: 10.1149/10701.4631ecst.
- [46] D.-J. Jwo and S.-F. Chiu, "Deep Learning Based Automated Detection of Diseases from Apple Leaf Images," *Computers, Materials & Continua*, vol. 71, no. 1, pp. 1849–1866, 2022, doi: 10.32604/cmc.2022.021875.
- [47] S. Abisha and T. Jayasree, "Application of Image Processing Techniques and Artificial Neural Network for Detection of Diseases on Brinjal Leaf," *IETE J Res*, vol. 68, no. 3, pp. 2246–2258, 2022, doi: 10.1080/03772063.2019.1696716.
- [48] J. Chen, D. Zhang, M. Suzaiddola, and A. Zeb, "Identifying crop diseases using attention embedded MobileNet-V2 model," *Appl Soft Comput*, vol. 113, p. 107901, 2021, doi: 10.1016/j.asoc.2021.107901.
- [49] A. Hemalatha and J. Vijayakumar, "Automatic Tomato Leaf Diseases Classification and Recognition using Transfer Learning Model with Image Processing Techniques," in *2021 Smart Technologies, Communication and Robotics (STCR)*, IEEE, 2021, pp. 1–5. doi: 10.1109/STCR51658.2021.9588993.
- [50] M. E. H. Chowdhury *et al.*, "Automatic and Reliable Leaf Disease Detection Using Deep Learning Techniques," *AgriEngineering*, vol. 3, no. 2, pp. 294–312, 2021, doi: 10.3390/agriengineering3020020.
- [51] W. Li, D. Wang, M. Li, Y. Gao, J. Wu, and X. Yang, "Field detection of tiny pests from sticky trap images using deep learning in agricultural greenhouse," *Comput Electron Agric*, vol. 183, p. 106048, 2021, doi: 10.1016/j.compag.2021.106048.
- [52] R. S. Latha *et al.*, "Automatic Detection of Tea Leaf Diseases using Deep Convolution Neural Network," in *2021 International Conference on Computer Communication and Informatics (ICCCI)*, IEEE, 2021, pp. 1–6. doi: 10.1109/ICCCI50826.2021.9402225.
- [53] A. A. Yattoo and A. Sharma, "A Novel Model for Automatic Crop Disease Detection," in *2021 Sixth International Conference on Image Information Processing (ICIIP)*, IEEE, 2021, pp. 310–313. doi: 10.1109/ICIIP53038.2021.9702553.
- [54] H. Nagar and R. S. Sharma, "Pest Detection on Leaf using Image Processing," in *2021 International Conference on Computer Communication and Informatics (ICCCI)*, IEEE, 2021, pp. 1–5. doi: 10.1109/ICCCI50826.2021.9402606.
- [55] A. K. Saini, R. Bhatnagar, and D. K. Srivastava, "AI based automatic detection of citrus fruit and leaves diseases using deep neural network model," *Journal of*

- Discrete Mathematical Sciences and Cryptography*, vol. 24, no. 8, pp. 2181–2193, 2021, doi: 10.1080/09720529.2021.2011095.
- [56] M. Z. U. Rehman *et al.*, “Classification of Citrus Plant Diseases Using Deep Transfer Learning,” *Computers, Materials & Continua*, vol. 70, no. 1, pp. 1401–1417, 2022, doi: 10.32604/cmc.2022.019046.
- [57] A. Khattak *et al.*, “Automatic Detection of Citrus Fruit and Leaves Diseases Using Deep Neural Network Model,” *IEEE Access*, vol. 9, pp. 112942–112954, 2021, doi: 10.1109/ACCESS.2021.3096895.
- [58] A. Afifi, A. Alhumam, and A. Abdelwahab, “Convolutional Neural Network for Automatic Identification of Plant Diseases with Limited Data,” *Plants*, vol. 10, no. 1, p. 28, 2020, doi: 10.3390/plants10010028.
- [59] Md. A. Iqbal and K. H. Talukder, “Detection of Potato Disease Using Image Segmentation and Machine Learning,” in *2020 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, IEEE, 2020, pp. 43–47. doi: 10.1109/WiSPNET48689.2020.9198563.
- [60] D. Argüeso *et al.*, “Few-Shot Learning approach for plant disease classification using images taken in the field,” *Comput Electron Agric*, vol. 175, p. 105542, 2020, doi: 10.1016/j.compag.2020.105542.
- [61] D. Das, M. Singh, S. S. Mohanty, and S. Chakravarty, “Leaf Disease Detection using Support Vector Machine,” in *2020 International Conference on Communication and Signal Processing (ICCSP)*, IEEE, 2020, pp. 1036–1040. doi: 10.1109/ICCSP48568.2020.9182128.
- [62] P. K. Mugithe, R. V. Mudunuri, B. Rajasekar, and S. Karthikeyan, “Image Processing Technique for Automatic Detection of Plant Diseases and Alerting System in Agricultural Farms,” in *2020 International Conference on Communication and Signal Processing (ICCSP)*, IEEE, 2020, pp. 1603–1607. doi: 10.1109/ICCSP48568.2020.9182065.
- [63] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanekharan, “Using deep transfer learning for image-based plant disease identification,” *Comput Electron Agric*, vol. 173, p. 105393, 2020, doi: 10.1016/j.compag.2020.105393.
- [64] E. C. Tetila *et al.*, “Automatic Recognition of Soybean Leaf Diseases Using UAV Images and Deep Convolutional Neural Networks,” *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 5, pp. 903–907, 2020, doi: 10.1109/LGRS.2019.2932385.
- [65] K. Gangadharan, G. Rosline, D. Dhanasekaran, and K. Malathi, “Automatic Detection of Plant Disease and Insect Attack using EFFT Algorithm,” *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 2, 2020, doi: 10.14569/IJACSA.2020.0110221.

- [66] M. M. Ozguven and K. Adem, "Automatic detection and classification of leaf spot disease in sugar beet using deep learning algorithms," *Physica A: Statistical Mechanics and its Applications*, vol. 535, p. 122537, 2019, doi: 10.1016/j.physa.2019.122537.
- [67] B. Doh, D. Zhang, Y. Shen, F. Hussain, R. F. Doh, and K. Ayepah, "Automatic Citrus Fruit Disease Detection by Phenotyping Using Machine Learning," in *2019 25th International Conference on Automation and Computing (ICAC)*, IEEE, 2019, pp. 1–5. doi: 10.23919/IconAC.2019.8895102.
- [68] K. Thenmozhi and U. S. Reddy, "Crop pest classification based on deep convolutional neural network and transfer learning," *Comput Electron Agric*, vol. 164, p. 104906, 2019, doi: 10.1016/j.compag.2019.104906.
- [69] A. Elhassouny and F. Smarandache, "Smart mobile application to recognize tomato leaf diseases using Convolutional Neural Networks," in *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, IEEE, 2019, pp. 1–4. doi: 10.1109/ICCSRE.2019.8807737.
- [70] Md. R. Howlader, U. Habiba, R. H. Faisal, and Md. M. Rahman, "Automatic Recognition of Guava Leaf Diseases using Deep Convolution Neural Network," in *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, IEEE, 2019, pp. 1–5. doi: 10.1109/ECACE.2019.8679421.
- [71] N. R. Bhimte and V. R. Thool, "Diseases Detection of Cotton Leaf Spot Using Image Processing and SVM Classifier," in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, 2018, pp. 340–344. doi: 10.1109/ICCONS.2018.8662906.
- [72] S. M. Jaisakthi, P. Mirunalini, D. Thenmozhi, and Vatsala, "Grape Leaf Disease Identification using Machine Learning Techniques," in *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*, IEEE, 2019, pp. 1–6. doi: 10.1109/ICCIDS.2019.8862084.
- [73] P. Tm, A. Pranathi, K. SaiAshritha, N. B. Chittaragi, and S. G. Koolagudi, "Tomato Leaf Disease Detection Using Convolutional Neural Networks," in *2018 Eleventh International Conference on Contemporary Computing (IC3)*, IEEE, 2018, pp. 1–5. doi: 10.1109/IC3.2018.8530532.
- [74] K. L. Roldán-Serrato, J. A. S. Escalante-Estrada, and M. T. Rodríguez-González, "Automatic pest detection on bean and potato crops by applying neural classifiers," *Engineering in Agriculture, Environment and Food*, vol. 11, no. 4, pp. 245–255, 2018, doi: 10.1016/j.eaef.2018.08.003.
- [75] J. Shijie, J. Peiyi, H. Siping, and sLiu Haibo, "Automatic detection of tomato diseases and pests based on leaf images," in *2017 Chinese Automation Congress (CAC)*, IEEE, 2017, pp. 2510–2537. doi: 10.1109/CAC.2017.8243388.

- [76] Y. Lu, S. Yi, N. Zeng, Y. Liu, and Y. Zhang, "Identification of rice diseases using deep convolutional neural networks," *Neurocomputing*, vol. 267, pp. 378–384, 2017, doi: 10.1016/j.neucom.2017.06.023.
- [77] V. Singh and A. K. Misra, "Detection of plant leaf diseases using image segmentation and soft computing techniques," *Information Processing in Agriculture*, vol. 4, no. 1, pp. 41–49, 2017, doi: 10.1016/j.inpa.2016.10.005.
- [78] S.-A. Raza, G. Prince, J. P. Clarkson, and N. M. Rajpoot, "Automatic Detection of Diseased Tomato Plants Using Thermal and Stereo Visible Light Images," *PLoS One*, vol. 10, no. 4, p. e0123262, 2015, doi: 10.1371/journal.pone.0123262.
- [79] H. Qi, Y. Liang, Q. Ding, and J. Zou, "Automatic Identification of Peanut-Leaf Diseases Based on Stack Ensemble," *Applied Sciences*, vol. 11, no. 4, p. 1950, 2021, doi: 10.3390/app11041950.
- [80] Plantix, "Plantix App." Accessed: Jun. 04, 2024. [Online]. Available: <https://plantix.net/en/>
- [81] PictureThis, "PictureThis - Botanist In Your Pocket." Accessed: Jun. 04, 2024. [Online]. Available: <https://www.picturethisai.com/>
- [82] P. S. N. Sarah, "Leaf Doctor: A New Portable Application for Quantifying Plant Disease Severity," *APS Publications*, Aug. 2015, Accessed: Jun. 04, 2024. [Online]. Available: <https://apsjournals.apsnet.org/doi/10.1094/PDIS-03-15-0319-RE>
- [83] AI Agronomist, "AI Agronomist." Accessed: Jun. 04, 2024. [Online]. Available: <https://www.aiagronomist.com/>
- [84] P. Gorbachenko, "What are Functional and Non-Functional Requirements and How to Document These," ENKONIX. Accessed: Mar. 12, 2024. [Online]. Available: <https://enkonix.com/blog/functional-requirements-vs-non-functional/>
- [85] G. Krüger and C. Lane, "How to Write a Software Requirements Specification (SRS Document)," PERFORCE. Accessed: Mar. 12, 2024. [Online]. Available: <https://www.perforce.com/blog/alm/how-write-software-requirements-specification-srs-document>
- [86] V. Paradigm, "What is Use Case Diagram," Visual Paradigm Website. Accessed: Mar. 15, 2024. [Online]. Available: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/>
- [87] T. Loton, "Data Modeling: Entity-Relationship Diagram (ER Diagram)," Modern Analyst. Accessed: Mar. 10, 2024. [Online]. Available: <https://www.modernanalyst.com/Resources/Articles/tabid/115/ID/2008/Data-Modeling-Entity-Relationship-Diagram-ER-Diagram.aspx>

- [88] V. Paradigm, "Comprehensive Guide to Entity-Relationship (ER) Modeling: With Example," VISUAL PARADIGM. Accessed: Mar. 12, 2024. [Online]. Available: <https://blog.visual-paradigm.com/comprehensive-guide-to-entity-relationship-er-modeling-with-example/>
- [89] A. Freeman, *Pro ASP.NET MVC 5*. Apress, 2014.
- [90] J. M. R. Lerman, *Programming Entity Framework: DbContext*. O'Reilly Media, 2012.
- [91] *SQL Server 2019 Revealed*. Apress, 2019.
- [92] Flutter Team, "Flutter Documentation," Flutter. Accessed: Apr. 17, 2024. [Online]. Available: <https://flutter.dev/docs>
- [93] Microsoft, "Visual Studio Code Documentation," Microsoft. Accessed: Apr. 17, 2024. [Online]. Available: <https://code.visualstudio.com/docs>
- [94] Google, "Teachable Machine," Google. Accessed: Apr. 17, 2024. [Online]. Available: <https://teachablemachine.withgoogle.com/>