Journal of Education Technology and Information System Vol (02) Ed (01) P.44-58 (2026)
Received June 2025/ Revised February 2026 / Accepted February 2026

Journal of Education Technology and Information System

https://journal.unesa.ac.id/index.php/jetis/

Design of An Android Application for Leaf Disease Detection in Plants

Muhammad Nizam Setiawan[1*], Ardhini Warih Utami[1], Fauzan Nusyura[2]

[1] *State University of Surabaya, Surabaya, Indonesia*
[2] *Universiti Kebangsaan Malaysia, Malaysia*
muhammad.21058@mhs.unesa.ac.id , ardhiniwarih@unesa.ac.id, P141316@siswa.ukm.edu.my

Abstract. Agriculture plays a strategic role in Indonesia's economy, with approximately 29,342,202 individual agricultural enterprises recorded in 2023, according to Statistics Indonesia (BPS). Golokan Village, located in Sidayu District, Gresik Regency, is an agrarian area where 23.22% of the population works as farmers, and it has a total agricultural land area of 385 hectares. However, between 2019 and 2023, production of three main commodities declined significantly: corn from 302.5 tons to 275.6 tons, tomatoes from 810 tons to 585 tons, and cassava from 1,000 tons to 832 tons. One contributing factor is the difficulty in detecting plant diseases early. To address this challenge, this study designed and developed an Android application called AgroAI that utilizes deep learning, specifically a Convolutional Neural Network (CNN) model based on the MobileNet architecture, optimized with TensorFlow Lite for mobile devices. The development was carried out using the Scrum methodology in two sprints. The first sprint included needs analysis, dataset collection, interface design, and model training. The second sprint implemented core features, including leaf disease detection via camera or gallery, classification results with recommended solutions, analysis history management, educational articles, and user authentication via Firebase. Black-box testing confirmed that all features functioned as intended, and model validation achieved an accuracy of 94.74%. This application is expected to enhance farmers' efficiency in crop management and support the sustainability of both local and national agricultural sectors.

Keywords: CNN, Deep Learning, Plant Disease Detection, Scrum, TensorFlow Lite

1. Introduction

As a sector that meets the main food needs, agriculture also provides livelihoods for a large portion of the population (Rojun & Nadziroh, 2020). According to data from the Central Bureau of Statistics (BPS) in 2023, there are approximately 29,342,202 individual agricultural enterprises in Indonesia, highlighting the dominance of this sector in the national economic structure (Lina Sudarwati & Nasution, 2024).

In general, the agricultural sector comprises several sub-sectors: food crops, horticulture, and plantations (Heldayani et al., 2022). Food crops such as corn play an important role as staple ingredients in meeting both local and national food needs (Malau et al., 2023). Horticultural crops like tomatoes are also leading commodities that support diversity in consumption and market demand. Meanwhile, cassava, which falls under plantation crops, has high economic value as a raw material for various processed products, such as tapioca flour and traditional foods (Natbais & Umbu, 2023)

One of the focus locations for this research is Golokan Village, located in Sidayu Sub-district, Gresik Regency, which is known as an agrarian area with significant agricultural potential. According to data from the official website of the Golokan Village Government, the majority of its population (23.22%) works as farmers, making agriculture the backbone of the village economy. The total agricultural land area is 385 hectares (Shobirin, 2020).

However, despite the agricultural sector's strengths in Golokan Village, farmers face various challenges that can hinder their productivity. Based on official data from the Golokan Village Farmers Group Association (Gapoktan), the total land area used for corn, tomatoes, and cassava cultivation ranged from 150 to 160 hectares from 2019 to 2023.

Agricultural production trends in Golokan Village over the past five years show a decline in harvest yields for the three main commodities: corn, tomatoes, and cassava. To provide a clearer picture of these production changes, the following diagram illustrates the fluctuation in crop yields from 2019 to 2023 (Figure 1).

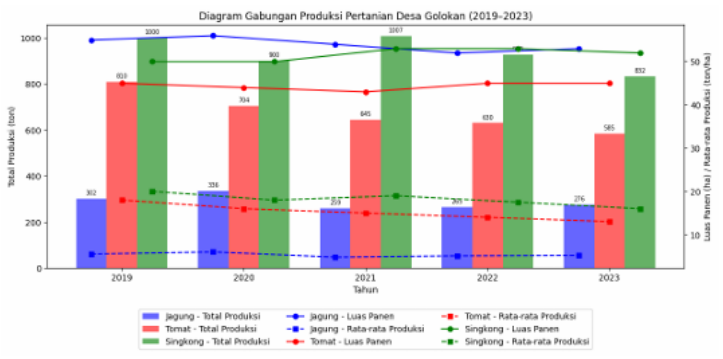


Figure 1 Village Agricultural Production Results Diagram

Based on the diagram above, the production of corn, tomatoes, and cassava in Golokan Village has declined year by year. Corn production dropped from 302.5 tons (2019) to 275.6 tons (2023), even though farmers can harvest it up to three times a year. Tomatoes experienced a significant decrease from 810 tons to 585 tons, despite being harvestable four times a year. Similarly, cassava declined from 1,000 tons to 832 tons, with a twice-yearly harvest cycle. This decline not only affects farmers' livelihoods but also threatens the sustainability of the region's agricultural sector (GAPOKTAN, 2025).

One of the main causes of this production decline is the emergence of various plant diseases. These diseases are marked by changes in all or part of the plant organs, disrupting daily physiological activities (Fuadi & Suharso, 2022). The causes vary, including infections by pathogens such as fungi, bacteria, viruses, and others (Iswantoro & Handayani UN, 2022). Therefore, early detection of plant diseases is crucial to maintain agricultural productivity and sustainability. Recent advancements in artificial intelligence have significantly improved image-based plant disease detection. A comprehensive review by (Sajitha, 2024) highlights that deep learning approaches, particularly convolutional neural networks (CNNs), have become the dominant method for plant disease classification due to their high accuracy and automation capability. However, challenges remain regarding dataset imbalance, computational complexity, and limited deployment in real-world mobile environments.

Recent studies have developed Android-based applications for plant disease detection using deep learning techniques. (Bhagat et al., 2024) demonstrated that lightweight deep learning models are essential for real-time plant disease detection on mobile and edge devices, as they significantly reduce computational cost while maintaining competitive classification performance. Furthermore, (Upadhyay et al., 2025) emphasized that future research should prioritize model efficiency, multi-crop generalization, and real-time mobile deployment to enhance practical adoption in rural agricultural environments. In addition to model efficiency, recent research has also highlighted the importance of explainable artificial intelligence (XAI) in plant disease detection systems. (Porfírio et al., 2025) reported that incorporating explainability techniques into convolutional neural network models enhances transparency, interpretability, and user trust, particularly in agricultural decisionsupport systems. Despite these advancements, many mobile-based plant disease detection applications still prioritize accuracy over interpretability.

Several application-oriented studies have also been conducted at the implementation level. (Sutisna and Soegoto, 2025) designed a potato pest and disease detection application using Teachable Machine integrated with TensorFlow Lite. Although the application demonstrated satisfactory detection performance and practical usability for farmers, the study did not provide detailed evaluation metrics such as precision, recall, and F1-score, limiting the assessment of classification robustness across classes. In addition, the system was limited to a single commodity (potatoes), limiting its broader agricultural applicability. (Yulita et al., 2023) implemented a DenseNet-based convolutional neural network for tomato leaf disease detection in a mobile application, achieving 95.7% accuracy using 10-fold cross-validation. While the study showed strong classification performance, the DenseNet architecture is relatively computationally intensive compared to lightweight models specifically designed for mobile deployment, and the research focused solely on tomato plants without extending the framework to multiple crop types within a unified system.

Similarly, (Khalil et al., 2024) developed an Android-based rice disease detection application using the Object-Oriented Analysis and Design (OOAD) methodology. The system achieved 80% detection accuracy and emphasized structured software development. However, the classification performance remained relatively moderate, and the study did not integrate a lightweight deep learning architecture optimized for real-time mobile inference.

Based on these studies, it is evident that although various plant disease detection applications have been developed, several gaps remain. Many studies focus on a single crop commodity, use computationally intensive architectures, or lack comprehensive evaluation metrics. Despite these advancements, no prior study has simultaneously addressed lightweight mobile deployment, multi-commodity classification, comprehensive evaluation metrics, and practical field validation within a unified Android-based system.

Accordingly, this study proposes the development of an Android-based application for detecting plant leaf diseases using a lightweight Convolutional Neural Network architecture optimized for mobile deployment. The application is designed to help farmers quickly, accurately, and conveniently identify disease types. In addition to disease detection, the system provides data management features to support monitoring and documentation of plant conditions. By adopting a user-oriented development methodology, the proposed solution aims to improve disease detection efficiency, reduce crop losses, and contribute to agricultural sustainability in Indonesia.

2. Methods

2.1 Research Design

This study employed a quantitative research approach with a case study conducted in Desa Golokan, Kecamatan Sidayu, Kabupaten Gresik. The objective was to design and implement an Android-based application for plant leaf disease detection using a Convolutional Neural Network (CNN) model integrated through TensorFlow Lite. The quantitative approach was used to evaluate model performance based on numerical metrics such as accuracy, precision, recall, and F1-score.

The system development process adopted the Scrum framework within the Agile methodology to ensure iterative, incremental software development. Meanwhile, the plant disease classification model was developed using the MobileNet architecture to achieve efficient performance on mobile devices.

2.2 Scrum-Based Application Development

Scrum is one of the approaches within the Agile methodology, first introduced by Ken Schwaber in 1997 (Utami et al., 2022) As part of Agile, Scrum offers significant value and benefits in software development by focusing on iterative cycles to develop innovative products or services (Robinson, 2024). In Scrum, there are three main roles: the Product Owner (PO), the Scrum Master, and the Development Team. The PO is responsible for product management, the Development Team is a cross-functional, self-organizing group, and the Scrum Master leads the team and helps remove obstacles to ensure software quality (Robinson, 2024).

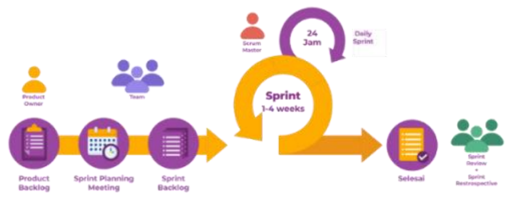


Figure 2 Scrum Development Model

In Figure 2, the principles of the Scrum process are illustrated. The workflow in Scrum begins with the entire team defining the Scrum building block known as a sprint. A sprint is a time-boxed period lasting between one and four weeks, during which the development team focuses on achieving specific targets. Each sprint ends with a review that showcasing the results from a team meeting. In Scrum, there are several important phases, namely:

A. Product Backlog

The Product Backlog is a list of requirements that contains detailed features to be developed, based on the specifications gathered during data collection.

B. Sprint Backlog

The Sprint Backlog is a list of items selected from the product backlog to be worked on during the sprint.

2.3 Dataset Description

The dataset used in this study consists of 24,572 leaf images categorized into 19 classes, including healthy and diseased leaves from tomato, corn, and cassava plants. The images were obtained from a combination of publicly available plant disease datasets and field documentation conducted in Golokan Village. Publicly sourced images followed their original annotations, while field-collected images were manually verified by experienced local farmers to ensure labeling accuracy and consistency.

The inclusion of field data captured under natural environmental conditions such as varying lighting, background complexity, and leaf orientation was intended to enhance real-world applicability and improve model generalization capability.

The dataset is distributed across three plant commodities with different numbers of classes and image counts, as summarized in Table 1.

Table 1 Dataset

Plant Type	Number of Classes	Total Images
Tomato	10	18,293
Corn	4	3,527
Cassava	5	2,752
Total	**19**	**24,572**

Tomato contributes 18,293 images divided into ten categories: Bacterial Spot (1,730 images), Early Blight (1,921), Two-Spot Spider Mite (1,742), Leaf Mold (1,883), Late Blight (1,852), Yellow Leaf Curl Virus (1,962), Septoria Leaf Spot (1,746), Mosaic Virus (1,703), Target Spot (1,828), and Healthy leaves (1,926).

Corn includes 3,527 images categorized into Northern Leaf Blight (1,000 images), Common Rust (1,014), Gray Leaf Spot (513), and Healthy leaves (1,000).

Cassava consists of 2,752 images divided into Cassava Bacterial Blight (421 images), Cassava Mosaic Disease (820), Cassava Green Mite (774), Cassava Brown Streak Disease (421), and Healthy leaves (316).

Overall, the dataset show class imbalance, tomato images dominate, while cassava image are relatively rare. Such an imbalance may influence model learning behavior, as deep learning models tend to favor majority classes. Therefore, evaluation metrics beyond overall accuracy, including precision, recall, and F1-score, were employed to ensure balanced performance assessment across categories.

The relatively large dataset size and multi-commodity coverage are expected to improve model robustness by providing diverse visual patterns of disease symptoms captured under varying environmental conditions.

2.4Data Preprocessing and Augmentation

All images were resized to 64 × 64 pixels with 3 RGB channels, and their pixel values were normalized to 0–1 to improve model convergence during training. The dataset was divided using stratified sampling into 80% training data (19,657 images) and 20% testing data (4,915 images), with a portion of the training data further allocated as validation data during training. Data augmentation was applied exclusively to the training set using the ImageDataGenerator library, inckude rotation up to 90 degrees and horizontal flipping. The validation and test datasets were not augmented to ensure objective and unbiased performance evaluation.

2.5CNN Model Architecture

The classification model was developed using MobileNet as the base model due to its lightweight structure and suitability for mobile deployment. MobileNet utilizes depthwise separable convolution to reduce computational cost while maintaining classification performance.

Additional layers were added to enhance feature extraction and classification performance, including a Conv2D layer with 32 filters and ReLU activation, a Dropout layer with a rate of 0.2, a Global Average Pooling layer, and a Dense layer with 19 output neurons using Softmax activation. The base MobileNet model was initialized with ImageNet pre-trained weights (include_top=False) to leverage transfer learning for improved feature extraction.

2.6Model Training Strategy

The model was trained using the Adam optimizer with categorical cross-entropy as the loss function for multi-class classification. The training configuration is summarized in Table 2.

Table 2 Model Training Configuration

Hyperparameter	Value
Batch Size	98
Epochs	100
Optimizer	Adam
Callbacks	EarlyStopping, ModelCheckpoint

EarlyStopping was implemented to prevent overfitting, while ModelCheckpoint was used to save the best-performing model based on validation loss. The model was trained using categorical cross-entropy loss due to the multi-class classification setting involving 19 classes.

2.7Model Evaluation

Model performance was evaluated using accuracy, precision, recall, and F1-score derived from the confusion matrix. Accuracy measures the proportion of correctly classified samples, while precision and recall evaluate per-class prediction reliability. The F1-score provides a harmonic balance between precision and recall to handle potential class imbalance.

3. Results and Discussion

3.1 Product Backlog

This list is compiled based on the results of identifying user needs and user stories in table 1

Table 3 Product Backlog List

ID	Backlog Item	Order of Priority
1	System requirements analysis and collection of plant disease datasets	1
2	Application interface design (UI/UX)	2
3	User Authentication	3
4	Select Image from Gallery or Camera	4
5	Show Classification Results with Labels and Solutions	5
6	History of Plant Disease Analysis	6
7	Profile Updates and Feedback Submission	7
8	Access Articles on Plant Diseases	8

3.1.1 Requirement Planning

In the initial stage of system development, problem identification and user requirements were identified through literature reviews, field observations, and interviews. Although the development method used is the iterative Scrum approach, requirements planning was still conducted to establish the initial backlog. The system requirements are classified into functional and non-functional needs. The functional requirements include the application's ability to classify leaf images into disease or healthy categories, allow users to select images from the camera or gallery, display classification results along with suggested solutions, provide login and registration features, manage analysis history, update user profiles, submit feedback, and access articles related to plant diseases. Meanwhile, the non-functional requirements cover optimal performance on mid-range Android devices, a responsive, user-friendly interface, local storage for classification data, image processing and classification results delivered in under 2 minutes, secure authentication using Firebase, and the ability to update articles and disease solution data from an external database.

3.1.2 Application Development

3.1.2.1 Use Case Diagram

A Use Case Diagram illustrates the interaction between users and the system in accomplishing specific processes or functions.

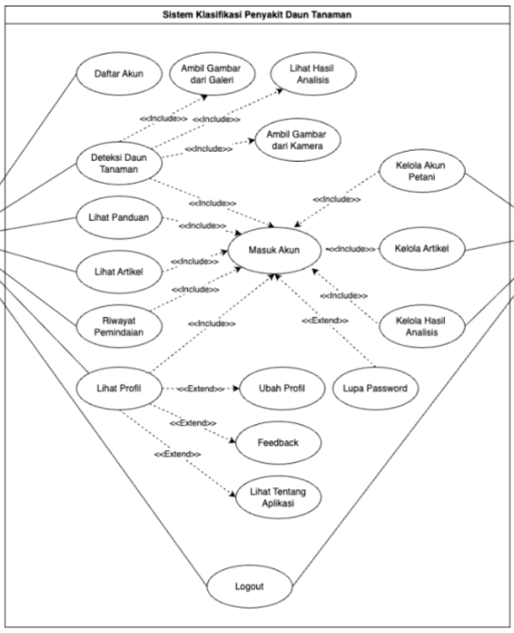
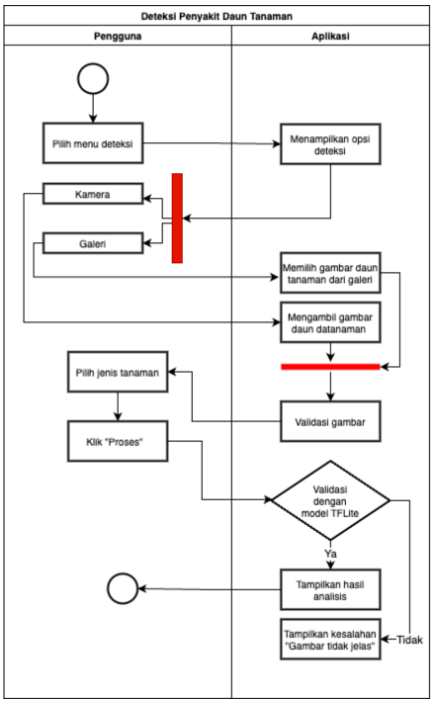


Figure 3 Use Case Diagram

3.1.2.2 Activity Diagram

The image below shows the steps for detecting plant leaf diseases. The user selects the detection menu, then the system displays options to use the camera or gallery. After selecting an image, the user is directed to the detection result page.


Deteksi Penyakit Daun Tanaman
Pengguna
Aplikasi
Pilih menu deteksi
Menampilkan opsi deteksi
Kamera
Galeri
Memilih gambar daun tanaman dari galeri
Mengambil gambar daun datanaman
Pilih jenis tanaman
Validasi gambar
Klik "Proses"
Validasi dengan model TFLite
Ya
Tampilkan hasil analisis
Tampilkan kesalahan "Gambar tidak jelas"
Tidak

Feature 4 Activity Diagram

3.1.2.3 Testing Phase

A. Application Development Testing

At this stage, the researcher conducted functional testing of the application using the black-box testing method, creating test cases to verify that outputs matched expectations. The testing involved six farmers with diverse backgrounds to evaluate features such as login, data input, and image classification. Results were marked as "Success" if they met expectations or "Fail" if they did not.

B. Plant Leaf Disease Validation Testing

Testing was conducted by importing images in two ways: either by capturing directly with the camera or by importing from the app's gallery. A total of 10 images were used for each plant disease type. Once the images were input, the system identified whether the tomato, corn, or cassava leaf was healthy or infected. Based on the entire testing process, the validation results were concluded using the following percentage calculation formula.

$$p = \frac{a}{t} \times 100$$

p = percentage of plant disease validation
a = number of correct/valid predictions for new inputs
t = total number of new inputs (valued at 10)

This validation formula refers to the approach used by Irawan et al., (2021), which calculates the accuracy of plant disease identification by comparing the number of correct predictions to the total number of test data point. This approach evaluates the performance of the Convolutional Neural Network (CNN) model embedded in the application against previously unseen image data.

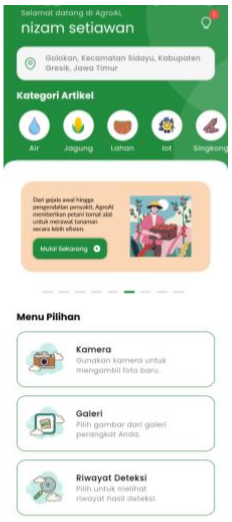
3.2 Sprint Backlog

3.2.1 Prototype

This application uses Firebase Firestore as its database with six main collections: Analyze, Articles, Banners, Categories, Feedback, and Users. The Flutter project structure is neatly organized into several folders, such as assets, bindings, common, styles, and widgets. The data folder contains repositories for CRUD operations on Firebase, while the features folder manages the app's main features, including controllers, models, and UI views, facilitating efficient development and maintenance.

3.2.1.1 Homepage

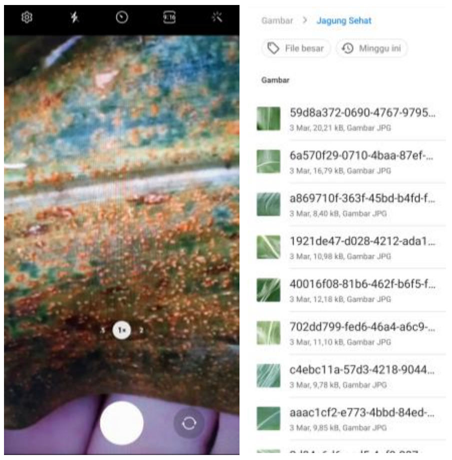
This page displays the application interface after login, where users can see their location information and four main menu options: Camera to take new photos for detecting plant diseases, Gallery to select images from the device, Detection History to view previous analysis results, and Articles that provide agricultural information, such as plant care and disease control.


Selamat datang di AgroAI,
nizam setiawan
Golokan, Kecamatan Sidayu, Kabupaten Gresik, Jawa Timur
Kategori Artikel
Air
Jagung
Lahan
Iot
Singkong
Dari gejala awal hingga pengendalian penyakit, AgroAI memberikan petani tomat alat untuk merawat tanaman secara lebih efisien.
Mulai Sekarang
Menu Pilihan
Kamera
Gunakan kamera untuk mengambil foto baru.
Galeri
Pilih gambar dari galeri perangkat Anda.
Riwayat Deteksi
Pilih untuk melihat riwayat hasil deteksi.

Feature 5 Home page

3.2.1.2 Image Analysis

This page outlines the process for capturing images of plant diseases using either the device's camera or the gallery. Users can choose the "Camera" option to directly take a photo of the plant part they want to inspect, or select "Gallery" to upload an image already stored on the device.

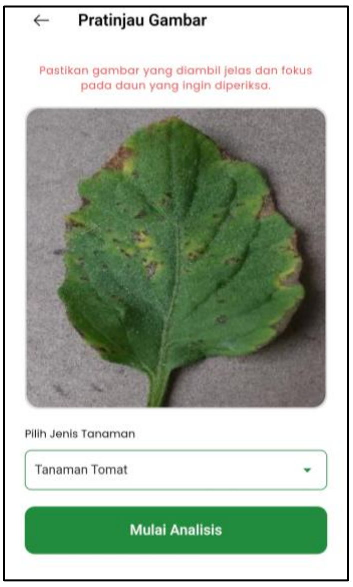


59d8a372-0690-4767-9795...
3 Mar, 20,21 kB, Gambar JPG
6a570f29-0710-4baa-87ef-...
3 Mar, 16,79 kB, Gambar JPG
a869710f-363f-45bd-b4fd-f...
3 Mar, 8,40 kB, Gambar JPG
1921de47-d028-4212-ada1...
3 Mar, 10,98 kB, Gambar JPG
40016f08-81b6-462f-b6f5-f...
3 Mar, 12,18 kB, Gambar JPG
702dd799-fed6-46a4-a6c9-...
3 Mar, 11,10 kB, Gambar JPG
c4ebc11a-57d3-4218-9044...
3 Mar, 9,78 kB, Gambar JPG
aaac1cf2-e773-4bbd-84ed-...
3 Mar, 9,85 kB, Gambar JPG

Feature 6 Image Analysis

3.2.1.3 Analysis Preview

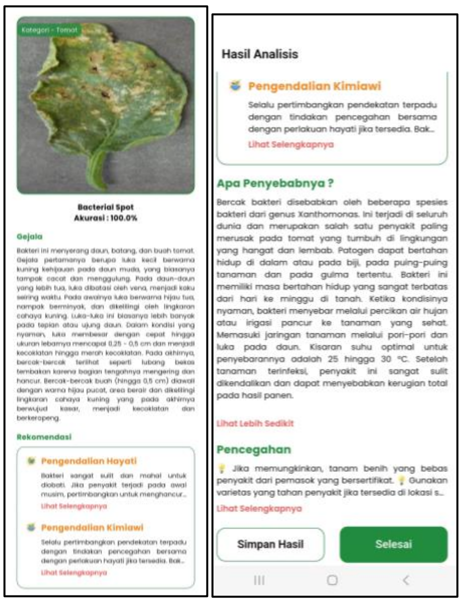
This page displays a preview of the image to be analyzed after the user takes a photo of the plant disease using the camera or selects an image from the gallery. Users are asked to ensure that the image is clear and focused on the part of the leaf they want to inspect. Additionally, you can select the type of plant, such as tomato, corn, or cassava. After confirming the image and selecting the plant type, users can press the "Start Analysis" button to begin the disease detection process.


Pratinjau Gambar
Pastikan gambar yang diambil jelas dan fokus pada daun yang ingin diperiksa.
Pilih Jenis Tanaman
Tanaman Tomat
Mulai Analisis

Feature 7 Analysis Preview

3.2.1.4 Analysis Results

This page displays the results of the plant disease analysis, including the disease name, detection accuracy level, and the type of plant analyzed (e.g., tomato). The information presented includes disease symptoms, control recommendations (both biological and chemical), disease causes, and preventive measures. Users can save the analysis results for future reference or complete the process by pressing the "Finish" button.


Kategori - Tomat

Bacterial Spot
Akurasi : 100.0%

Gejala

Bakteri ini menyerang daun, batang, dan buah tomat. Gejala pertamanya berupa luka kecil berwarna kuning kehijauan pada daun muda, yang biasanya tampak cacat dan menggulung. Pada daun-daun yang lebih tua, luka dibatasi oleh vena, menjadi kaku seiring waktu. Pada awalnya luka berwarna hijau tua, nampak berminyak, dan dikelilingi oleh lingkaran cahaya kuning. Luka-luka ini biasanya lebih banyak pada tepian atau ujung daun. Dalam kondisi yang nyaman, luka membesar dengan cepat hingga ukuran lebarnya mencapai 0.25 - 0,5 cm dan menjadi kecoklatan hingga merah kecoklatan. Pada akhirnya, bercak-bercak terlihat seperti lubang bekas tembakan karena bagian tengahnya mengering dan hancur. Bercak-bercak buah (hingga 0,5 cm) diawali dengan warna hijau pucat, area berair dan dikelilingi lingkaran cahaya kuning yang pada akhirnya berwujud kasar, menjadi kecoklatan dan berkeropeng.

Rekomendasi

Pengendalian Hayati

Bakteri sangat sulit dan mahal untuk diobati. Jika penyakit terjadi pada awal musim, pertimbangkan untuk menghancur...

Lihat Selengkapnya

Pengendalian Kimiawi

Selalu pertimbangkan pendekatan terpadu dengan tindakan pencegahan bersama dengan perlakuan hayati jika tersedia. Bak...

Lihat Selengkapnya

Hasil Analisis

Pengendalian Kimiawi

Selalu pertimbangkan pendekatan terpadu dengan tindakan pencegahan bersama dengan perlakuan hayati jika tersedia. Bak...

Lihat Selengkapnya

Apa Penyebabnya ?

Bercak bakteri disebabkan oleh beberapa spesies bakteri dari genus Xanthomonas. Ini terjadi di seluruh dunia dan merupakan salah satu penyakit paling merusak pada tomat yang tumbuh di lingkungan yang hangat dan lembab. Patogen dapat bertahan hidup di dalam atau pada biji, pada puing-puing tanaman dan pada gulma tertentu. Bakteri ini memiliki masa bertahan hidup yang sangat terbatas dari hari ke minggu di tanah. Ketika kondisinya nyaman, bakteri menyebar melalui percikan air hujan atau irigasi pancur ke tanaman yang sehat. Memasuki jaringan tanaman melalui pori-pori dan luka pada daun. Kisaran suhu optimal untuk penyebarannya adalah 25 hingga 30 °C. Setelah tanaman terinfeksi, penyakit ini sangat sulit dikendalikan dan dapat menyebabkan kerugian total pada hasil panen.

Lihat Lebih Sedikit

Pencegahan

💡 Jika memungkinkan, tanam benih yang bebas penyakit dari pemasok yang bersertifikat. 💡 Gunakan varietas yang tahan penyakit jika tersedia di lokasi s...

Lihat Selengkapnya

Feature 8 Analysis Results

3.3 Testing

3.3.1 Application Development Testing

Based on the results of the functional testing of the plant disease identification application, it can be concluded that all system functions work properly and as expected. The testing was conducted with six respondents, all farmers who are end users of the application. All test scenarios designed were successfully executed, and all features functioned as intended without any failures. Furthermore, the test results were calculated using the functional evaluation formula.

$$X = 1 - \frac{A}{B}$$

with:
X = System functionality level value
A = Number of failed functions
B = Total number of tested functions
If A=0 and B=1

$$X = 1 - \frac{0}{312} = 1$$

Based on the calculation results, the system functionality is valued at 1, indicating that the application has fulfilled this aspect with a very good rating.

3.3.2 Application Validation Testing

In the application validation stage, 6 farmers tested the application by entering 10 sample leaf images of each plant type independently and comparing the application's classification results with the actual conditions. The test results were used to calculate classification accuracy, and the application scored 180 out of 190 points. with the following calculations:

$$\frac{180}{190} \times 100 = 94.74\%$$

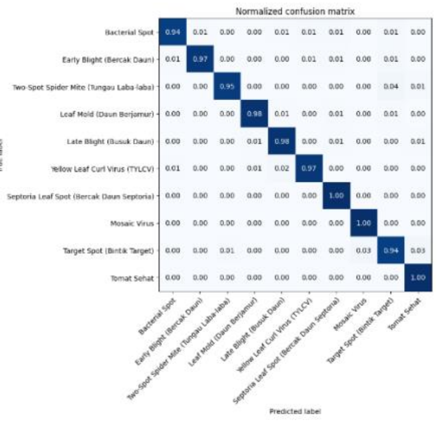
It can be concluded that the system's accuracy rate reached 94.74%, indicating that the application developed with the CNN model meets the criteria for excellent performance in classifying plant leaf diseases. This validation uses a formula based on the approach by Irawan et al. (2021), which calculates the accuracy of disease identification by comparing the number of correct predictions to the total number of test data points. It is important to distinguish between dataset-based evaluation and field validation testing. The confusion matrix results were obtained from the stratified test dataset and reflect the model's generalization performance across all samples. In contrast, the 94.74% validation accuracy was derived from limited real-world testing using selected representative images evaluated by farmers. Because the field validation involved a smaller, controlled sample, the results may differ from full-test-set statistical metrics and should be interpreted as preliminary usability validation rather than large-scale performance evaluation.

3.4 Evaluation

To comprehensively evaluate the performance of the proposed CNN-based plant leaf disease detection system, model evaluation was conducted using confusion matrices, classification metrics, functional testing, and real-world validation.

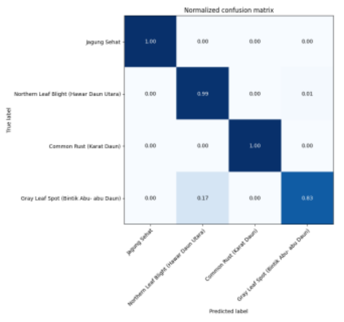
3.4.1 Confusion Matrix Analysis

To analyze classification performance in detail, confusion matrices were generated for the tomato, corn, and cassava datasets. The confusion matrix provides insight into the distribution of correct and incorrect predictions for each class and allows identification of misclassification patterns beyond overall accuracy.



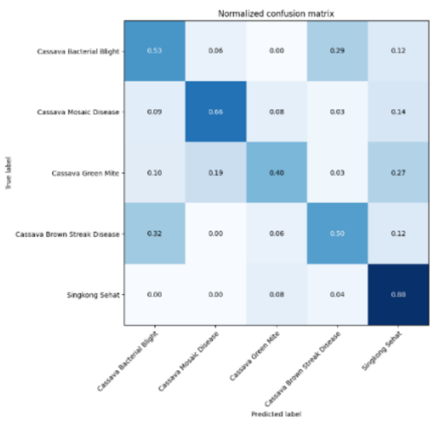
Feature 9 Normalized confusion matrix for tomato leaf disease classification.

The tomato model demonstrates strong classification performance across all ten classes. Most classes achieved high recall values, with Septoria Leaf Spot, Mosaic Virus, and Healthy Tomato reaching perfect recall (1.00). The overall classification accuracy for tomato diseases reached 97%, indicating robust feature extraction and strong class separation.



Feature 10 Normalized confusion matrix for corn leaf disease classification.

For corn classification, the model also achieved an overall accuracy of 97%. The Healthy Corn and Common Rust classes achieved perfect recall (1.00), while Northern Leaf Blight reached 0.99. However, minor misclassification occurred in Gray Leaf Spot, with approximately 12% of samples misclassified as Northern Leaf Blight. This confusion may be attributed to visual similarity between disease symptoms, which share overlapping leaf discoloration patterns.



Feature 11 Normalized confusion matrix for cassava leaf disease classification.

In contrast, the cassava model performed poorly, with an overall accuracy of 57%. Although the Healthy Cassava class achieved a recall of 0.88, several disease categories showed confusion with visually similar symptoms. The lower performance is likely due to the smaller dataset size and higher intra-class similarity, which reduces the model's generalization capability.

Overall, the confusion matrix analysis indicates that the model performs consistently well on tomato and corn datasets, while cassava classification remains a challenge due to data imbalance and symptom overlap.

3.4.2 Comparative Performance Summary

The overall classification performance across plant types is summarized in Table 4.

Tabel 4 Comparative Performance Summary

Plant Type	Accuracy	Precision (Macro)	Recall (Macro)	F1-Score (Macro)
Tomato	0.97	0.97	0.97	0.97
Corn	0.97	0.97	0.95	0.96
Cassava	0.57	0.57	0.59	0.55

3.4.3 Comparison with Other Methods

To further evaluate the effectiveness of the proposed MobileNet-based CNN model, a comparison was conducted with several related studies that implemented different architectures and development approaches for plant disease detection in mobile environments.

Table 5 compares the proposed method with previous studies on model architecture, commodity coverage, and classification performance.

Table 5 Comparison with Previous Studies

Study	Architecture	Commodity	Accuracy	Evaluation Metrics
Sutisna & Soegoto (2025)	Teachable Machine (TensorFlow Lite)	Potato	90%	Limited metrics
Yulita et al. (2023)	DenseNet	Tomato	95.70%	Accuracy, F1-score

Study	Architecture	Commodity	Accuracy	Evaluation Metrics
Khalil et al. (2024)	CNN-based (OOAD system)	Rice	80%	Accuracy
Proposed Method	MobileNet (Transfer Learning)	Tomato, Corn, Cassava (19 classes)	94.74% (overall), up to 97% per crop	Accuracy, Precision, Recall, F1-score, Confusion Matrix

Based on Table 5, the proposed MobileNet-based model achieves competitive classification performance compared to previous studies. Yulita et al. (2023) reported a slightly higher accuracy of 95.70% using a DenseNet architecture. However, DenseNet is known to be computationally intensive due to its dense connectivity pattern, which increases the number of parameters and memory requirements. In contrast, MobileNet utilizes depthwise separable convolution, significantly reducing computational cost while maintaining high classification performance. This makes the proposed method more suitable for deployment on mid-range Android devices with limited processing capabilities.

Compared to Sutisna and Soegoto (2025), who implemented Teachable Machine integrated with TensorFlow Lite and achieved 90% accuracy, the proposed system demonstrates improved classification robustness by providing comprehensive evaluation metrics, including precision, recall, F1-score, and confusion matrix analysis. The inclusion of detailed performance metrics enables better assessment of class-level prediction reliability.

Furthermore, Khalil et al. (2024) achieved 80% accuracy in rice disease detection using a CNN-based approach within an OOAD framework. Although their study emphasized structured software development, the classification performance remained moderate. The higher accuracy obtained with the proposed method may be attributed to the use of transfer learning with ImageNet-pretrained weights and to systematic training strategies such as early stopping and model checkpointing.

Another key distinction lies in commodity coverage. While previous studies focused on a single crop commodity, the proposed system integrates multi-commodity classification (tomato, corn, and cassava) within a unified mobile application consisting of 19 classes. This broader scope enhances the system’s practical applicability in rural agricultural environments where farmers cultivate multiple crop types simultaneously.

Overall, the comparison indicates that the proposed MobileNet-based approach achieves a strong balance between classification accuracy, computational efficiency, and practical usability, making it well-suited for real-time plant disease detection in mobile agricultural applications.

3.4.4 Model Limitations and Error Analysis

Although the proposed MobileNet-based model demonstrates strong classification performance, several limitations were identified during evaluation. Error analysis based on the confusion matrix reveals that misclassifications primarily occurred in classes with visually similar disease symptoms. For example, in the cassava dataset, several disease categories were confused due to overlapping leaf discoloration patterns and texture similarities. This indicates that the model may struggle to distinguish fine-grained visual differences when inter-class variation is minimal.

Another limitation relates to dataset imbalance. The dataset distribution is dominated by tomato images, while cassava has a relatively smaller number of samples. This imbalance may have contributed to the lower classification accuracy observed in cassava (57%). Models trained on imbalanced datasets tend to favor the majority classes, thereby reducing generalization to underrepresented categories.

In addition, environmental variation may influence model performance. Images collected from public datasets and field documentation may differ in lighting conditions, camera quality, background complexity, and leaf orientation. Such variability introduces potential dataset bias, which may affect real-world deployment performance when images are captured under different environmental conditions.

Furthermore, although the system was validated by six farmers, large-scale field testing across diverse geographic regions has not yet been conducted. Therefore, the current evaluation may not fully represent broader agricultural conditions. Future work should include expanding the dataset, improving class balance, incorporating a wider range of environmental samples, and conducting large-scale field validation to enhance model robustness and generalization.

4. Conclusions

Based on the research results, it can be concluded that the Android-based application AgroAI for classifying plant leaf diseases using TensorFlow Lite was successfully designed and developed using the Scrum methodology in two sprint stages. The application implements a MobileNet-based CNN that identifies 19 leaf condition classes, including healthy leaves, across corn, tomato, and cassava plants.

The development process encompassed requirement analysis, dataset collection and preprocessing, user interface design, CNN model training, and implementation of key features such as image-based disease detection via camera or gallery, classification results with treatment recommendations, analysis history management, article access, and Firebase-based user authentication.

Evaluation results demonstrated strong classification performance, particularly for tomato and corn datasets, while cassava classification requires further improvement. Functional testing confirmed that all application features operated as intended, and validation testing achieved an overall accuracy of 94.74%, indicating that the proposed system is both accurate and practically applicable for assisting farmers, especially in Golokan Village, in detecting plant leaf diseases through digital images.

For future development, improvements may focus on expanding the dataset, particularly for cassava, to improve class balance and generalization. The integration of model explainability techniques, such as Grad-CAM, is also recommended to provide visual interpretation of classification results and increase user trust. Additionally, expanding support to additional crop commodities and optimizing offline inference performance would further strengthen the system's applicability in rural agricultural environments. To support broader generalization, future field validation is recommended to involve farmers from multiple agricultural regions across different districts or provinces in Indonesia, with diverse environmental conditions, including varying soil types, climate zones, and farming practices. Collaboration with local agricultural extension services or farmer cooperatives (Gapoktan) at the national level could facilitate large-scale data collection and real-world performance evaluation, providing stronger empirical evidence for the system's applicability beyond the Golokan Village context.

References

Bhagat, S., Sharma, P., & Kumar, R. (2024). Advancing real-time plant disease detection: A lightweight deep learning approach for mobile and edge devices. *Smart Agricultural Technology, 6*, 100312. https://doi.org/10.1016/j.atech.2024.100312

Fuadi, A., & Suharso, A. (2022). Perbandingan arsitektur mobilenet dan nasnetmobile untuk klasifikasi penyakit pada citra daun kentang. *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *7*(3), 701–710.

GAPOKTAN. (2025). *Pertanian Desa Golokan.* Pemerintah Desa Golokan.

Heldayani, E., Asiyah, S., & Mardianto. (2022). Implementasi Metode Location Quotient (LQ) untuk Analisis Potensi Komoditas Unggulan Subsektor Hortikultura di Kabupaten Muara Enim.

Geodika: Jurnal Kajian Ilmu Dan Pendidikan Geografi, *6*(2), 220–231. https://doi.org/10.29408/geodika.v6i2.6496

Irawan, F., Sudarma, M., & Khrisne, D. (2021). Rancang Bangun Aplikasi Identifikasi Penyakit Tanaman Pepaya California Berbasis Android Menggunakan Metode Cnn Model Arsitektur Squeezenet. *Jurnal SPEKTRUM*, *8*(2), 18–27.

Iswantoro, D., & Handayani UN, D. (2022). Klasifikasi Penyakit Tanaman Jagung Menggunakan Metode Convolutional Neural Network (CNN). *Jurnal Ilmiah Universitas Batanghari Jambi*, *22*(2), 900–905. https://doi.org/10.33087/jiubj.v22i2.2065

Khalil, W., Irsan, M., & Fathoni, M. F. (2024). Designing an application for detecting diseases of rice plants using OOAD method. *Sinkron: Jurnal dan Penelitian Teknik Informatika, 8*(2).

Lina Sudarwati, & Nasution, N. F. (2024). Upaya Pemerintah dan Teknologi Pertanian dalam Meningkatkan Pembangunan dan Kesejahteraan Petani di Indonesia. *Jurnal Kajian Agraria Dan Kedaulatan Pangan (JKAKP)*, *3*(1), 1–8. https://doi.org/10.32734/jkakp.v3i1.15847

Malau, L. R. E., Rambe, K. R., Ulya, N. A., & Purba, A. G. (2023). Dampak perubahan iklim terhadap produksi tanaman pangan di indonesia. *Jurnal Penelitian Pertanian Terapan*, *23*(1), 34–46. https://doi.org/10.25181/jppt.v23i1.2418

Natbais, Y. H., & Umbu, A. B. S. (2023). Aplikasi Deteksi Penyakit pada Daun Tomat Berbasis Android Menggunakan Model Terlatih Tensorflow Lite. *TEKNOTAN*, *17*(2), 83–90. https://doi.org/10.24198/jt.vol17n2.1

Porfírio, R. P., Almeida, J., & Costa, L. (2025). Explainable artificial intelligence techniques for plant disease classification using convolutional neural networks. *Procedia Computer Science, 225*, 1450–1458. https://doi.org/10.1016/j.procs.2025.02.180

Robinson. (2024). *Agile Software Development.* TechTarget. https://www.techtarget.com/.

Rojun, M., & Nadziroh, N. (2020). Peran Sektor Pertanian Dalam Pertumbuhan Ekonomi Di Kabupaten Magetan The Role Of The Agricultural Sector In Economic Growth In Magetan Distric. In *Jurnal AGRISTAN* (Vol. 2, Issue 1).

Sajitha, P. (2024). A review on machine learning and deep learning image-based plant disease detection and classification systems. *Artificial Intelligence in Agriculture, 9*, 45–60. https://doi.org/10.1016/j.aiia.2024.01.004

Shobirin, M. (2020). *Persepsi Masyarakat terhadap Sifat Toksik Pestisida yang Berdampak pada Kesehatan Masyarakat dan Lingkungan di Desa Golokan, Kecamatan Sidayu, Kabupaten Gresik*. Universitas Brawijaya.

Sutisna, F. P., & Soegoto, B. K. (2025). Pest detection application development Android based diseases of potato plants. *International Journal of Research and Applied Technology, 5*(1), 251–255.

Upadhyay, A., Singh, V., & Sharma, D. (2025). Deep learning and computer vision in plant disease detection: Recent advances and future directions. *Artificial Intelligence Review*. https://doi.org/10.1007/s10462-024-11100

Utami, S., Eviyanti, K., Sari, W., & Haryanti, S. (2022). Rancang Bangun Aplikasi Edukasi Tuberkulosis Menggunakan Metode Scrum. *JURNAL INOVTEK POLBENG - SERI INFORMATIKA*, *7*(1), 83–96.

Yulita, I. N., Amri, N. A., & Hidayat, A. (2023). Mobile application for tomato plant leaf disease detection using a Dense Convolutional Network architecture. *Computation, 11*(2), 20.