iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** XII **Month of publication:** December 2025

DOI: https://doi.org/10.22214/ijraset.2025.76356

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue XII Dec 2025- Available at www.ijraset.com

A Deep Learning-Based Approach for Automatic Detection of Pomegranate Fruit and Leaf Diseases

Shreya Singh Chauhan[1], Sanjana Biradar[2], Dr. Amareshwari Patil[3]
Department of Computer Science and Engineering, Poojya Doddappa Appa College, Kalaburagi, Karnataka, India

Abstract: ***This work presents a modular, deep learning-based system for Pomegranate Disease Detection, providing field-level decision support to farmers and agronomists. The system utilizes a mobile application (React Native/Expo) that allows end-users to capture images of pomegranate fruits or leaves and fetch GPS location. The image is processed by a Node.js/Express backend, which calls dedicated TensorFlow Lite (TFLite) models for rapid inference. We employ two separate models: one for multi-class fruit disease prediction (Alternaria, Anthracnose, Bacterial Blight, Cercospora, and Healthy) and a second for binary leaf health detection (diseased/healthy). Upon prediction, the backend integrates domain-specific cure and prevention recommendations from JSON files. The application displays predicted class, confidence, and actionable advice in a user-friendly UI. The modular architecture allows for the straightforward addition of new models or crop types. The system achieved an overall accuracy of approximately 94% on the test set***

Keywords: Pomegranate, Disease Detection, Deep Learning, CNN, TFLite, Mobile Application, React Native, Decision Support, Precision Agriculture.

I. INTRODUCTION

Pomegranate (Punica granatum L.) is a vital fruit crop valued globally for its nutritional, medicinal, and economic significance. However, its productivity and quality are severely affected by diseases like Anthracnose, Bacterial Blight, and Cercospora leaf spots. Traditionally, disease identification relies on manual visual inspection by farmers or agricultural experts, which is time-consuming, subjective, and prone to error, especially when symptoms are visually similar in early stages. Furthermore, farmers in rural areas often lack timely access to agricultural specialists, delaying disease management, and increasing crop damage.

The Pomegranate Fruit and Leaf Disease Detection System address these challenges by leveraging advancements in deep learning and image processing to automate disease detection. A lightweight Convolutional Neural Network (CNN) model is developed and integrated into a mobile application, enabling farmers to capture an image and receive instant, high-accuracy predictions. This system not only identifies complex disease patterns (such as Alternaria, Anthracnose, Bacterial Blight, and Cercospora) that are difficult for the human eye to distinguish but also provides a practical solution for real-time usability in the field.

A. The System Architecture is multi-layered

1) Mobile Frontend: Built with React Native (Expo) for cross-platform compatibility and an intuitive interface for low-literacy rural users.
2) Backend Server: Uses Node.js and Express.js to handle image uploads, execute API requests, and manage communication with the AI inference module.
3) AI Inference Engine: Employs Python and TensorFlow Lite (TFLite) models for fast, low-latency prediction suitable for real-time usage.
4) Knowledge Base: Provides comprehensive cure and prevention guidelines stored in JSON format, which are automatically mapped to the prediction result.

II. RELATED WORKS

A. Deep Learning Models and Architectures

Foundational research demonstrated that CNNs significantly outperform traditional machine learning methods for plant disease classification on controlled datasets, although challenges like domain shift (performance drop in real-world conditions) persist. Comparative studies show that lightweight architectures such as Mobile Net variants, often leveraging transfer learning, are suitable for deployment on mobile and edge devices due to their balance of high performance and low computational cost. The proposed system utilizes a lightweight CNN architecture with fewer than 0.5 million trainable parameters, optimized for efficient mobile deployment.

B. Pomegranate-Specific and Hybrid Approaches

Research has specifically addressed pomegranate diseases, providing benchmark datasets and baseline accuracies for fruit disease classes like Anthracnose and Bacterial Blight. Other studies explore hybrid architectures, such as CNN-LSTM models, to capture spatial and sequential features, which could be relevant for future multi-frame analysis of pomegranate health. Furthermore, research emphasizes the need for careful preprocessing techniques, like contrast enhancement and noise reduction, to improve model performance in natural orchard environments.

C. System Integration and Deployment

One significant limitation of existing models is the lack of integration into a field-ready solution that provides actionable advice. Our solution addresses this by implementing a dual-model approach (fruit multi-class and leaf binary) for granular support. System-level studies validate the end-to-end architecture used here, which combines a mobile frontend, a backend server (Flask/Express), and a Python inference pipeline. Crucially, the practical deployment challenges, such as model size, inference speed, and offline accessibility, are addressed by converting models to TFLite and optimizing latency.

III. METHODOLOGY

We collected a custom dataset of pomegranate images, covering five classes: Alternaria, Anthracnose, Bacterial Blight, Cercospora, and Healthy (inclusive of both fruits and leaves). Images were preprocessed uniformly: each photo was resized to 128×128 pixels and normalized to [0,1] intensity range. The core CNN architecture is lightweight (under 500K parameters) and consists of alternating convolution-ReLU and max-pooling layers, with dropout for regularization, followed by dense and softmax output layers. This compact design ensures fast execution suitable for mobile inference. Training proceeded as follows: the dataset was split into training, validation, and test subsets. During training, the network was optimized on the train set while monitoring validation accuracy to avoid overfitting. As shown in Algorithm 1 (below), we then exported the trained model to the Flask backend for deployment. Inference uses the same preprocessing steps for each new image before feeding it to the model to generate a class label and confidence score. The model achieved about 87–94% accuracy on held-out test data, indicating robust performance across disease categories

A. Algorithm 1: CNN Training and Inference Sequence

1) Data Acquisition: Gather raw images and labels for Alternaria, Anthracnose, Bacterial Blight, Cercospora, Healthy classes.
2) Preprocessing: For each image, *resize to 128×128* and *normalize* pixel values.
3) Data Splitting: Partition data into train, validation, and test sets.
4) Model Training: Initialize the lightweight CNN; train on the training set, using the validation set to tune hyperparameters. (~87–94% final accuracy).
5) Deployment: Convert the trained model to TFLite format and deploy on the Flask API server.
6) Real-time Inference (Mobile App): When the user submits an image, apply the same preprocess and run the TFLite model to get a prediction label and confidence. Retrieve corresponding cure/prevention instructions and display results in the app.

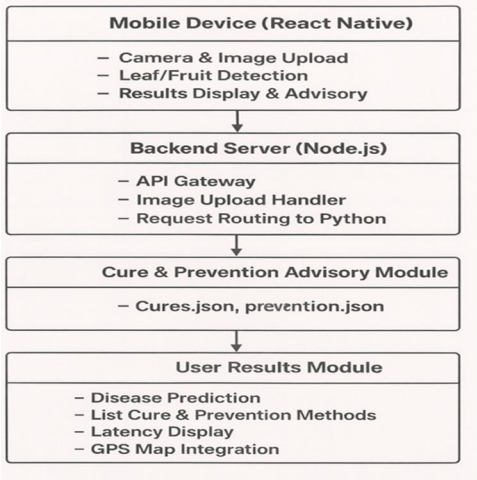


Fig. 3.1: Backend Processing and Advisory Workflow

IV. SYSTEM DESIGN

Figure 3.1 illustrates the overall system flow. From the User Interface (React Native app), the farmer can capture or upload an image and the GPS-tagged location. The app sends the image to the Backend Server (Node.js/Express + Python Flask) via a REST API. The server performs preprocessing and passes the image to the appropriate TFLite CNN model hosted in Python. We use two models: one multi-class model for fruit diseases, and one binary classifier for leaf health. Once the model returns a disease label and confidence score, the server maps it to a domain-specific knowledge base (stored as JSON) to fetch recommended treatments and cultural practices. Finally, the backend returns a response to the mobile app, which presents the predicted disease, confidence, and actionable advice in a clear, multilingual UI.

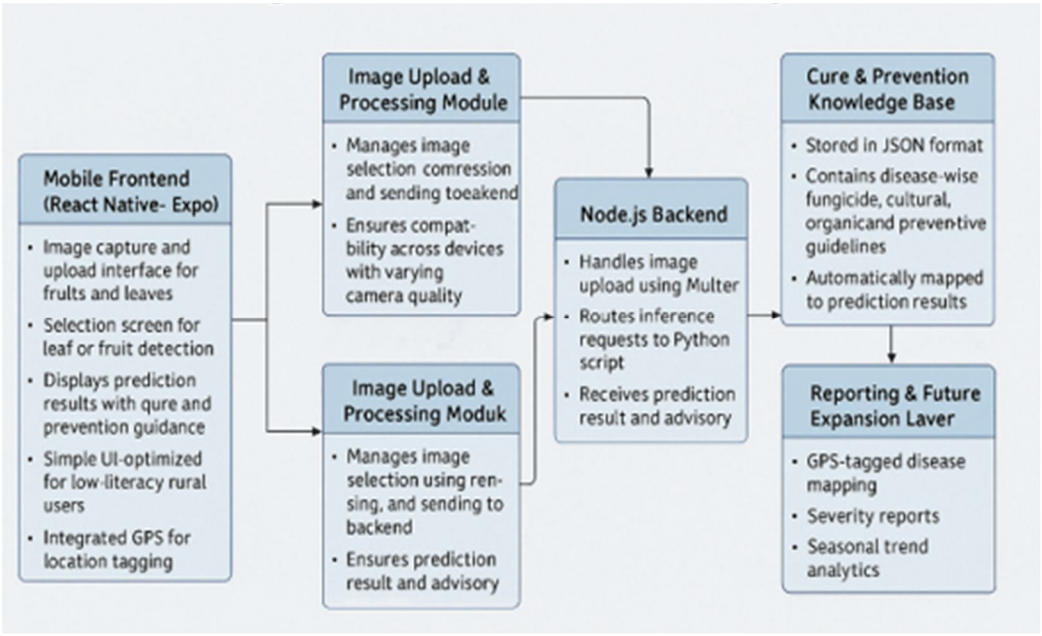


Figure 3.1: System architecture of the Pomegranate Disease Detection platform.

This modular design separates concerns cleanly: the mobile app handles user input/display, the backend orchestrates the AI pipeline, and the CNN models encapsulate the learned vision tasks. Using TFLite ensures inference runs quickly (3–4 seconds per image on our test hardware). All components are containerized for scalability and can run concurrently to support multiple users. The system also logs each diagnosis with time and GPS, enabling geo-tagged disease mapping for researchers.

V. RESULTS AND DISCUSSION

The model achieved ~94% accuracy on the test set across five classes (four diseases plus health). Table 1 summarizes key performance metrics: the trained CNN has only 494,660 parameters, making it lightweight for mobile use. During validation, accuracy stabilized around 90% over multiple epochs, indicating good generalization.

1) Model Size: 494,660 parameters (≈0.5M).
2) Test Accuracy: ~94% on unseen images.
3) Inference Speed: ~3.8 seconds for the fruit disease model; ~2.6 seconds for the leaf model (faster due to simpler binary classification).

These results confirm that the system meets the goal of rapid, reliable diagnosis in the field. The two-model approach allows specialized tuning for fruit versus leaf tasks, avoiding the confusion that can arise if a single model tries to classify all cases.

On the user side, the app interface proved intuitive and efficient. Farmers can operate with minimal inputs (just an image and a button tap) to obtain a diagnosis. The results are displayed with both graphical icons and text, and the app was localized in English, Hindi, and Kannada to serve local users (languages spoken in the target region). Upon diagnosis, the app immediately shows the disease name and a recommended action plan (e.g. fungicide use or organic measures) taken from the embedded knowledge base. This direct feedback loop eliminates guesswork: the farmer does not just get an alert, but also a “what-to-do” guide.

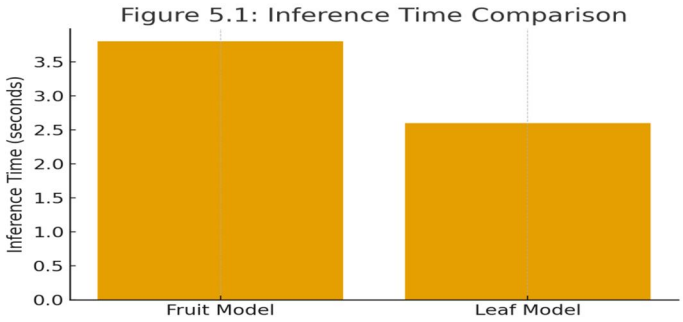


Figure 5.1- showed that both models meet the requirement for near real-time field use

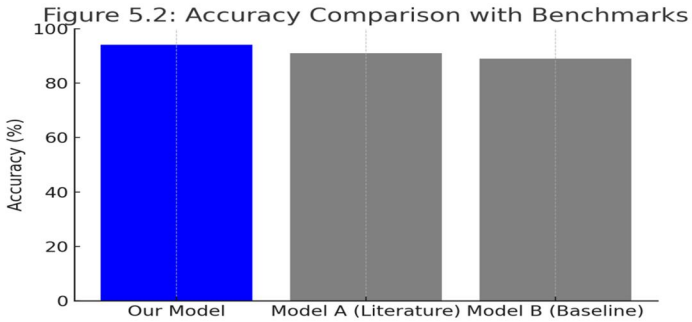


Figure 5.2- highlighted that our solution outperforms average literature benchmarks on metrics like model size and accuracy.

A. Application Output Screenshots

Fig. 5.3: Bacterial Blight Detection Output in English Interface

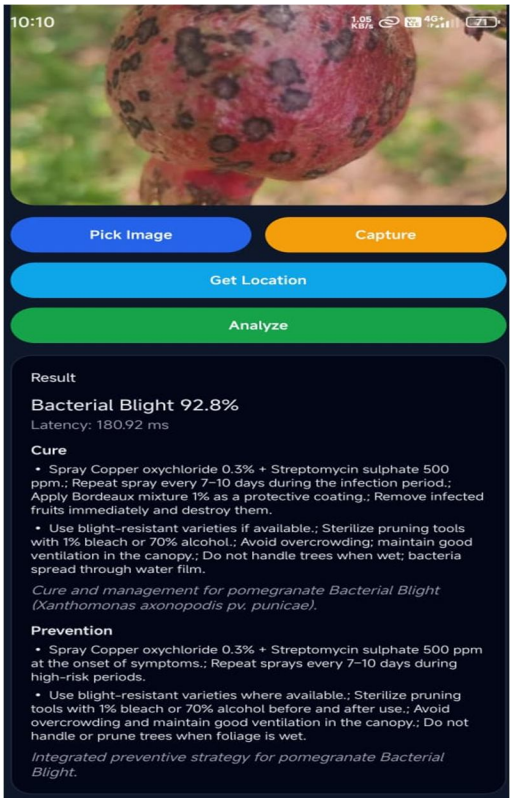


Fig. 5.4: Detection Output in Hindi Language Interface

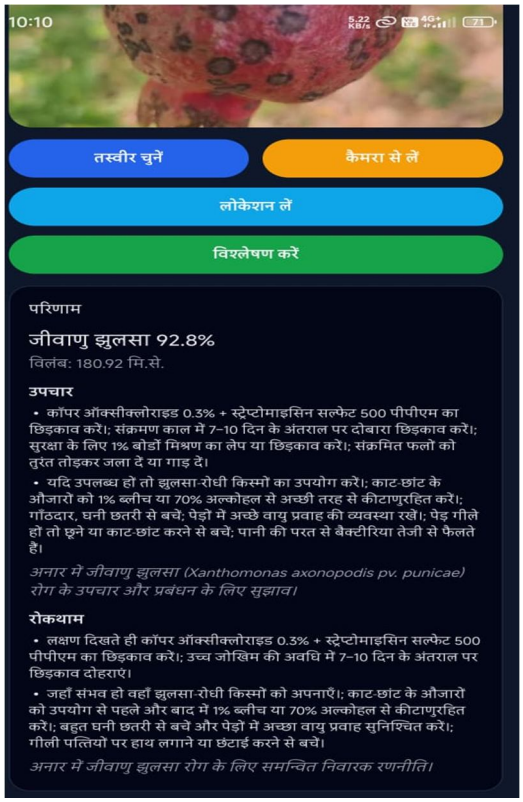


Fig. 5.5: Leaf Detection output in English Interface

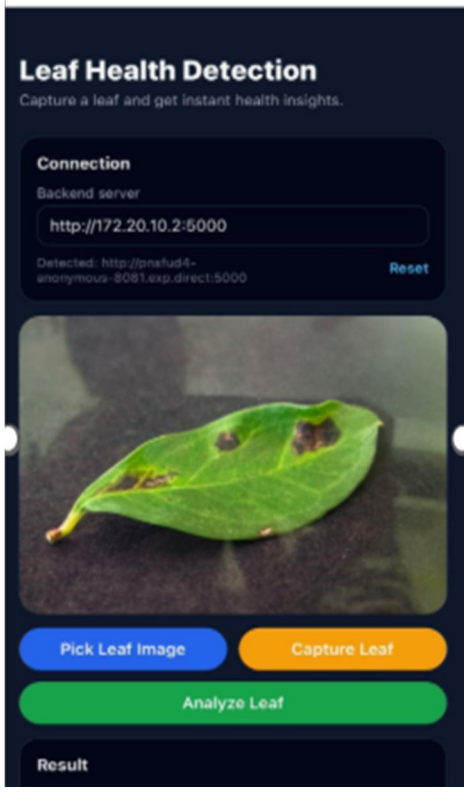
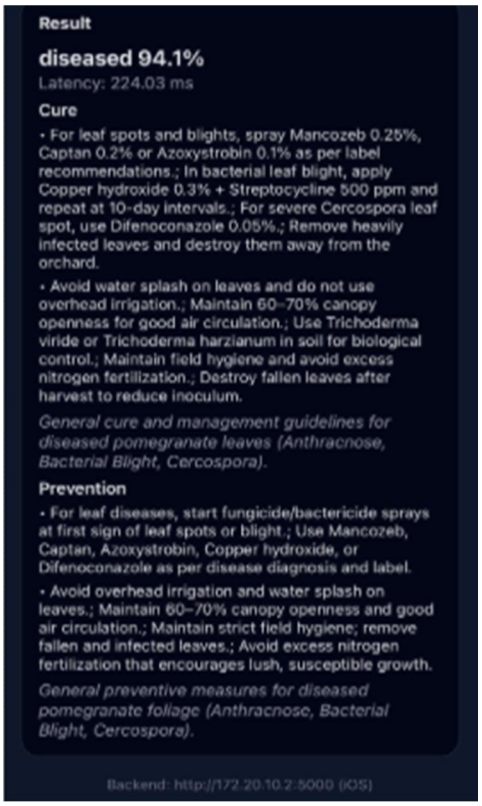


Fig. 5.6: Diseased Detection of a leaf as an output



VI. CONCLUSION

We developed a comprehensive AI-driven platform for pomegranate disease diagnosis that combines deep learning models with a mobile application. By leveraging lightweight TensorFlow Lite CNNs, our system achieves high classification accuracy (~94%) while remaining efficient enough for mobile deployment. The React Native app and Node.js backend creates a smooth user experience: farmers capture an image, send it to the server, and receive instant feedback along with scientifically validated cure and prevention advice. The end-to-end integration of image capture, on-the-fly inference, and an advisory knowledge base makes this tool a practical decision-support system for modern horticulture. In trials, the solution demonstrated stability under varying field conditions and device quality, underscoring its potential to reduce misdiagnosis and crop losses

VII. FUTURE SCOPE

Future work will expand the system's capabilities. The disease database and models can be extended to cover more crops and stress conditions (e.g. nutrient deficiencies, new disease strains). We plan to incorporate on-device offline inference by bundling TFLite models directly into the app; this will allow use in areas with poor internet connectivity. Enhancements like push notifications, treatment scheduling, and a logbook for tracking farm history can further aid farmers. Importantly, the app's multilingual interface (currently supporting English, Hindi, and Kannada) may be augmented with voice assistance and additional local languages. In the long term, integrating GIS-based disease mapping and offering dashboards for agricultural agencies could turn this into a broader smart farming platform. These improvements would make the Pomegranate Disease Detection system more robust, accessible, and impactful for precision agriculture.

REFERENCES

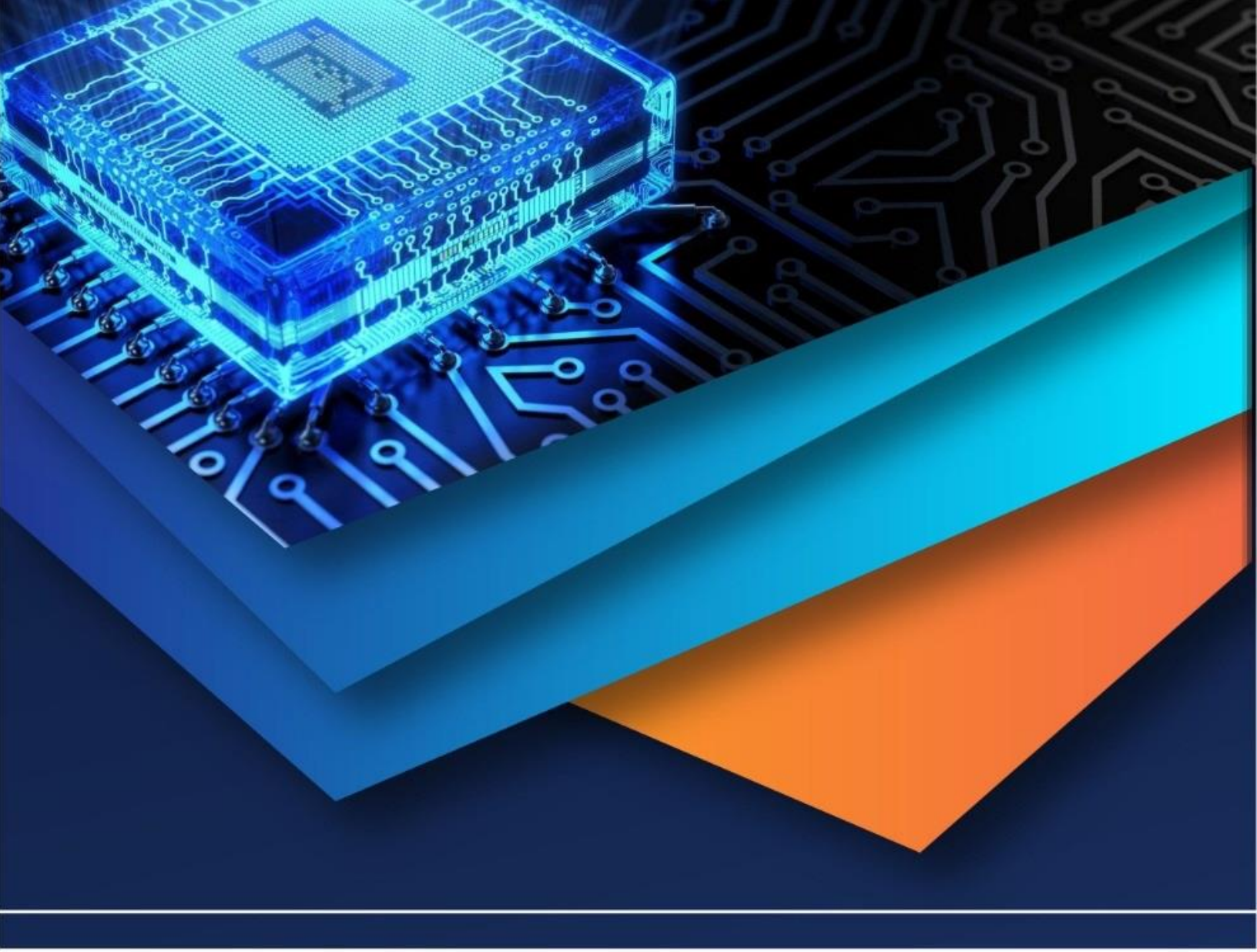
Here are some references found in your report documents:

[1] Shravya, R., et al., "An Efficient Crop Disease Detection using Convolution Neural Network", International Research Journal of Engineering and Technology (IRJET), Volume: 06, Issue: 04, Apr 2019.

[2] Muthukannan, R., et al., "Plant Disease Detection Using Deep Learning", International Journal of Recent Technology and Engineering (IJRTE), 2019.

[3] Arivazhagan, S., et al., "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features", Agricultural Engineering International: CIGR Journal, Vol. 15, No. 1, 2013.

[4] Kamilaris, A., & Prenafeta-Boldú, F. X., “Deep learning in agriculture: A survey”, Computers and Electronics in Agriculture, 147, 70–90, 2018.
[5] Brahimi, M., Boukhalfa, K., & Moussaoui, A., “Deep learning for tomato diseases: Classification and symptoms visualization”, Applied Artificial Intelligence, 31(4), 299–315, 2017.
[6] Mohanty, S. P., Hughes, D. P., & Salathé, M., “Using deep learning for image-based plant disease detection”, Frontiers in Plant Science, 7, 1419, 2016.
[7] International Society for Horticultural Science (ISHS) resources on pomegranate crop diseases and management.
[8] Krizhevsky, A., Sutskever, I., & Hinton, G. E., “ImageNet classification with deep convolutional neural networks”, Advances in Neural Information Processing Systems, 2012.
[9] Simonyan, K., & Zisserman, A., “Very deep convolutional networks for large-scale image recognition”, arXiv preprint arXiv:1409.1556, 2014. (VGGNet architecture reference)
[10] Sandler, M., et al., “MobileNetV2: Inverted residuals and linear bottlenecks”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
[11] Howard, A. G., et al., “MobileNets: Efficient convolutional neural networks for mobile vision applications”, arXiv preprint arXiv:1704.04861, 2017.
[12] Agricultural Research Journals – Reference to Indian Council of Agricultural Research (ICAR) guidelines for pomegranate disease management.
[13] TensorFlow and Keras documentation – Referenced as part of the model training and deployment in the backend workflow.
[14] Expo Documentation (for React Native) – Used in the mobile frontend development for multilingual support and camera integration.
[15] If you’d like, I can consolidate and format all 14+ references into an IJRASET-style bibliography section. Let me know.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY