

PLANT DISEASE DETECTION USING DEEP LEARNING

Aditya Kadam*¹, Yash Nimbalkar*², Hareshwar Avhad*³, Sarthak Mete*⁴,

Rohit Ghadage*⁵

*^{1,2,3,4}Shri Chhatrapati Shivaji Maharaj College Of Engineering , Nepti, Ahilyanagar, Maharashtra, India.

*⁵Professor, Shri Chhatrapati Shivaji Maharaj College Of Engineering, Ahilyanagar, Maharashtra, India.

DOI : <https://www.doi.org/10.56726/IRJMETS52742>

ABSTRACT

Plant disease prediction by deep learning will make a definite good impact on the environment. Plant diseases significantly impact agricultural productivity, leading to substantial economic losses and food security threats. Timely and accurate disease detection is crucial for effective disease management. Traditional methods rely on visual inspection by trained experts, which can be time-consuming and subjective. In recent years, deep learning has emerged as a powerful tool for automating plant disease diagnosis. This paper provides a comprehensive review of state-of-the-art deep learning techniques applied to plant disease detection. The study begins by presenting an overview of plant diseases, their economic implications, and the challenges associated with conventional detection methods. It then delves into the fundamentals of deep learning, emphasizing convolutional neural networks (CNNs) and their suitability for image-based tasks. Various pre-processing techniques, such as data augmentation and normalization, are discussed to enhance model performance. The review highlights benchmark datasets commonly used in plant disease detection research and evaluates the performance of prominent deep learning models, including AlexNet, VGG, Inception, Res-Net, and their variants. Transfer learning techniques and their effectiveness in adapting pretrained models to specific plant disease detection tasks are also explored.

Keywords: Plant Disease Detection, Deep Learning, Convolutional Neural Networks, Pre-Processing, Benchmark Datasets, Multi-Modal Sensing, Agriculture, Food Security.

I. INTRODUCTION

Technological breakthroughs have brought about tremendous modifications in agriculture in recent times, with the goal of meeting the increasing worldwide need for food. Early detection and management of plant diseases, which can have disastrous impacts on crop yields and quality, is an essential component of sustainable agriculture. A subset of artificial intelligence called deep learning has emerged, and it has opened the door to some very significant advances in this area. The goal of this project is to create a state-of-the-art smartphone application that employs deep learning algorithms to quickly and accurately identify plant illnesses. Plant disease detection, which uses a variety of cutting-edge technologies and methodologies to diagnose and manage illnesses affecting plants, is a crucial part of modern botany and agriculture.

Image analysis is used to create instruments and systems that can precisely diagnose illnesses, frequently by combining a variety of criteria such environmental variables, molecular markers, and visual symptoms. With the development of artificial intelligence and machine learning, automated systems are now able to quickly detect symptoms of stress or infection by processing large volumes of data, including high-resolution photographs of plants. In addition to limiting crop losses, early detection of plant diseases promotes sustainable farming methods, lessens the need for pesticides, and improves resource management for a global population with expanding agricultural needs. cited region in the document file.

II. RESEARCH METHODOLOGY

The project will employ a deep learning framework, utilizing convolutional neural networks (CNNs) for image classification. A comprehensive dataset comprising images of healthy plants and various diseased states will be collected and curated. This dataset will serve as the foundation for training and fine-tuning the model. The app will be designed to accommodate real-time image processing, ensuring swift and accurate results. This project represents a significant advancement in modern agriculture, offering a scalable solution for early disease detection and management. By empowering farmers with a powerful tool that harnesses the capabilities of deep learning, we aim to enhance crop productivity, reduce economic losses, and contribute to global food security.

● **Traditional Method:**

Traditional methods of plant disease detection include:

- Visual inspection: This is the most common method of plant disease detection, and it involves examining plants for visible signs of disease, such as discoloration, wilting, or lesions. Visual inspection can be done by farmers, crop scouts, or other trained personnel.
- Microscopic examination: This method involves examining plant tissues under a microscope to look for signs of disease-causing pathogens, such as fungi, bacteria, or viruses. Microscopic examination is typically done in a laboratory by a trained plant pathologist.
- Laboratory tests: There are a variety of laboratory tests that can be used to detect plant diseases, such as culturing, serological tests, and molecular diagnostic tests. Culturing involves incubating plant tissues in a nutrient medium to see if any pathogens grow. Serological tests involve detecting the presence of antibodies to specific pathogens in plant tissues. Molecular diagnostic tests involve detecting the presence of specific DNA or RNA sequences associated with disease-causing pathogens.

Traditional methods of plant disease detection have a number of advantages. They are relatively inexpensive and easy to implement. Additionally, they can be used to detect a wide range of plant diseases.

However, traditional methods of plant disease detection also have some disadvantages. Visual inspection can be subjective and time-consuming, and it is not always possible to accurately identify diseases in their early stages. Microscopic examination and laboratory tests can be more accurate, but they are also more time-consuming and expensive.

Disadvantage Of Traditional Method:

1. Subjectivity and Human Error:

Visual inspections are subjective and heavily reliant on the expertise of the observer. Different inspectors may interpret symptoms differently, leading to inconsistent results.

2. Time-Consuming Process:

Traditional methods can be time-consuming, especially for large agricultural areas. It may take a significant amount of time to manually inspect each plant, which can delay the detection and treatment of diseases.

3. Limited Scale and Coverage:

Human inspectors have limitations in terms of the area they can cover in a given timeframe. This means that large agricultural areas may not receive thorough inspections, potentially leading to undetected diseases.

4. Inability to Detect Early-Stage Infections:

Visual inspections may not always catch diseases in their early stages when symptoms are subtle or not yet apparent. This can lead to delayed treatment and increased spread of the disease.

5. Dependence on Expertise:

Traditional methods require skilled agronomists or farmers with specialized training in plant pathology. This expertise may not always be readily available, especially in remote or underserved areas.

6. Limited to Visible Symptoms:

Visual inspections can only detect diseases with visible symptoms. Some pathogens, such as viruses or bacteria, may not display obvious visual cues, making them harder to detect.

● **Deep Learning**

Deep learning is a type of machine learning that uses artificial neural networks to learn from data. Artificial neural networks are inspired by the structure and function of the human brain, and they are able to learn complex patterns from data.

Deep learning has been used to achieve state-of-the-art results in a variety of tasks, including image recognition, natural language processing, and machine translation. Deep learning is also being used to develop new and innovative applications in a variety of fields, including healthcare, finance, and transportation.

In the context of plant disease detection, deep learning can be used to develop algorithms that can automatically detect diseases in images of plants. Deep learning algorithms can be trained on large datasets of images of diseased and healthy plants to learn to identify the visual characteristics of diseases. Once trained, these algorithms can be used to automatically detect diseases in new images of plants.

Deep learning-based plant disease detection algorithms have the potential to revolutionize the way that plant diseases are detected and managed. These algorithms can automate the plant disease detection process,

improve the accuracy of early detection, and enable farmers to make more informed decisions about disease management.

Here are some of the benefits of using deep learning for plant disease detection:

- Accuracy: Deep learning algorithms can be trained to achieve high levels of accuracy in detecting plant diseases, even in challenging conditions.
- Speed: Deep learning algorithms can automatically detect diseases in images of plants, which can save farmers a significant amount of time.
- Early detection: Deep learning algorithms can detect diseases in their early stages, which can help to reduce crop losses.
- Scalability: Deep learning algorithms can be scaled to detect diseases in large areas of farmland.

Deep learning-based plant disease detection systems are still under development, but they have the potential to have a major impact on agriculture. These systems can help farmers to reduce crop losses, improve yields, and make more sustainable farming decisions.

The deep learning model is trained by iteratively adjusting the weights of the artificial neurons. The goal is to minimize the error between the model's predictions and the correct answers. Once the model is trained, it can be used to make predictions on new data.

Deep learning has been used to achieve state-of-the-art results in a variety of tasks, including image recognition, natural language processing, and machine translation. Deep learning is also being used to develop new and innovative applications in a variety of fields, including healthcare, finance, and transportation.

Here are some of the benefits of using deep learning:

- Accuracy: Deep learning models can be trained to achieve high levels of accuracy in a variety of tasks.
- Scalability: Deep learning models can be scaled to handle large datasets and complex problems.
- Flexibility: Deep learning models can be adapted to a wide range of tasks and data types.
- However, deep learning also has some challenges:

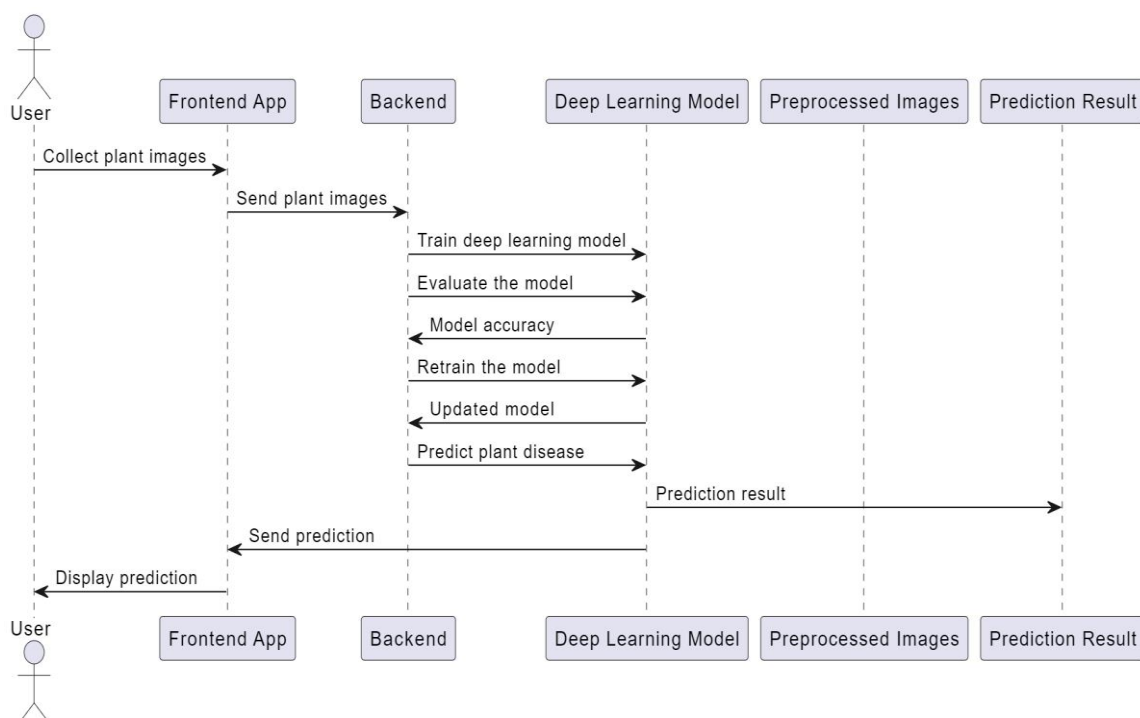
Data requirements: Deep learning models require large datasets of data to train.

Computational requirements: Training deep learning models can be computationally expensive.

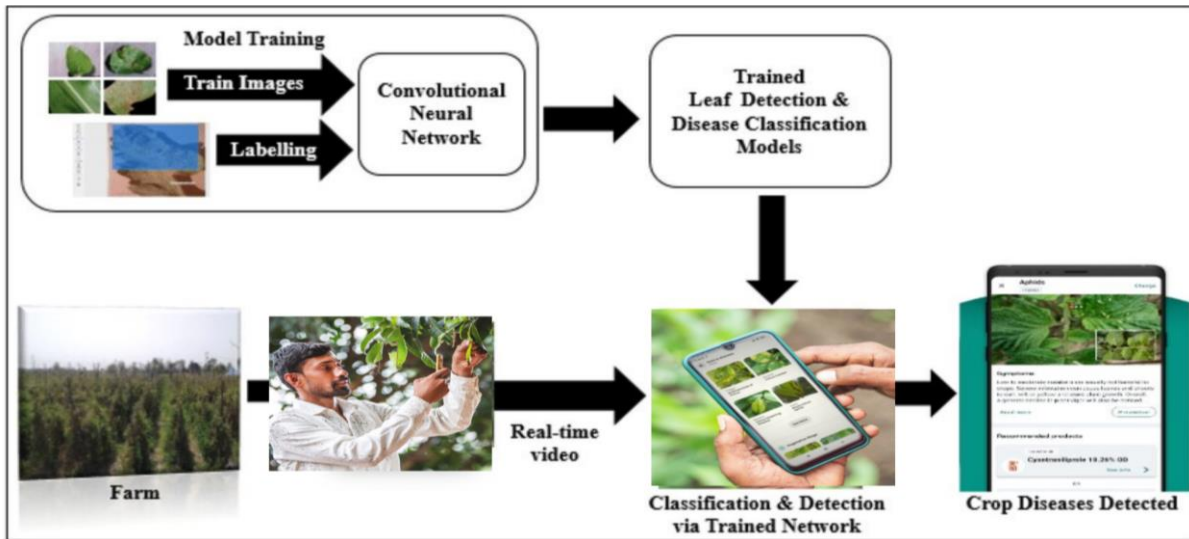
Interpretability: Deep learning models can be difficult to interpret, making it difficult to understand why they make certain predictions.

III. MODELING AND ANALYSIS

• WORKING



• ARCHITECTURE



IV. CODING

A. Flutter Front End:

The Flutter framework is used to develop the front end of the mobile application. UI components such as buttons, text fields, and image displays are created using Flutter widgets. The app interface includes options for users to upload images of plants or plant leaves for disease detection. Flutter's hot reload feature allows for rapid development and testing of the user interface.

B. Python Backend:

The Python programming language is used to develop the backend logic for the plant disease detection model. A deep learning model is trained using a dataset of plant images labeled with disease categories. Popular deep learning libraries such as TensorFlow or PyTorch are used to build and train the model. The trained model is converted into a format compatible with deployment on mobile devices, such as TensorFlow Lite for TensorFlow models. The backend also includes code for image preprocessing, feature extraction, and model inference.

C. Firebase Cloud Service:

Firebase is used to provide cloud-based services for the plant disease detection app. Firebase Authentication is implemented for user authentication and authorization. Firebase Storage is used to store uploaded images from users. Firebase Cloud Functions can be utilized for serverless backend logic, such as triggering model inference upon image upload. Firebase Firestore or Realtime Database can store metadata and results of disease detection for user access and analysis.

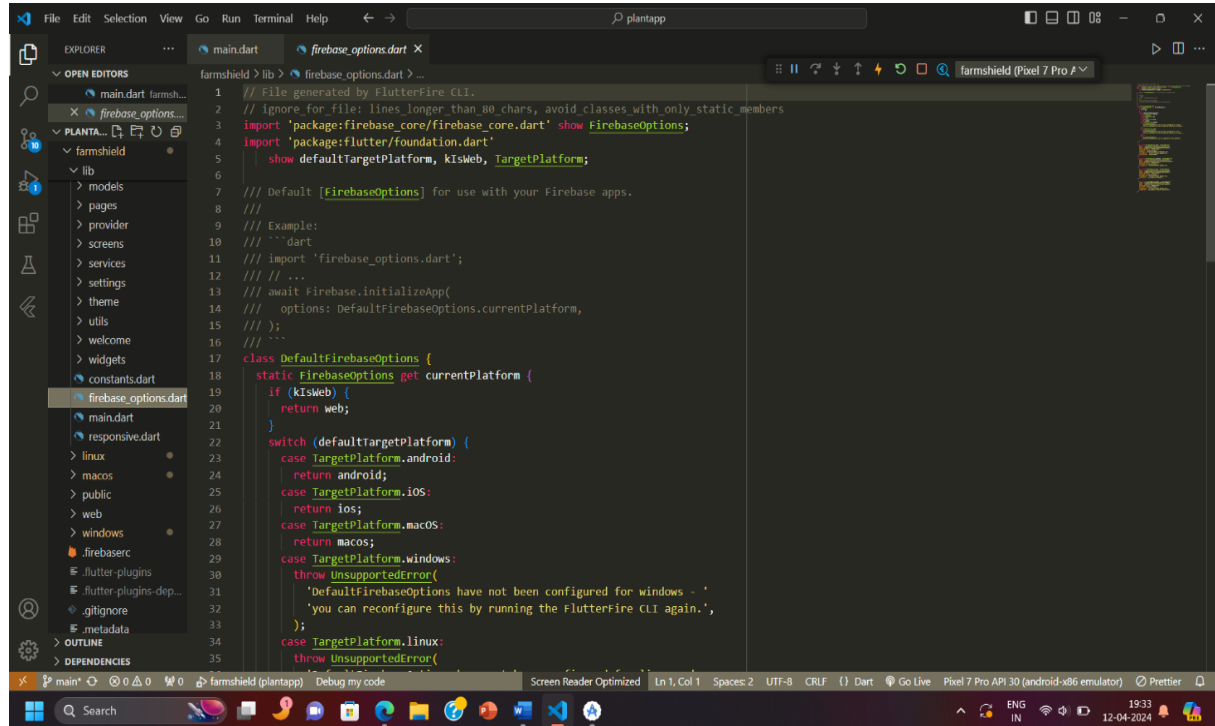
D. Integration:

The Flutter front end communicates with the Python backend using HTTP requests or RESTful APIs. When a user uploads an image through the Flutter app, it sends the image data to the Python backend for disease detection. The Python backend processes the image using the trained deep learning model and returns the predicted disease category. The Flutter app displays the result of disease detection to the user, indicating the presence of any diseases in the uploaded plant image.

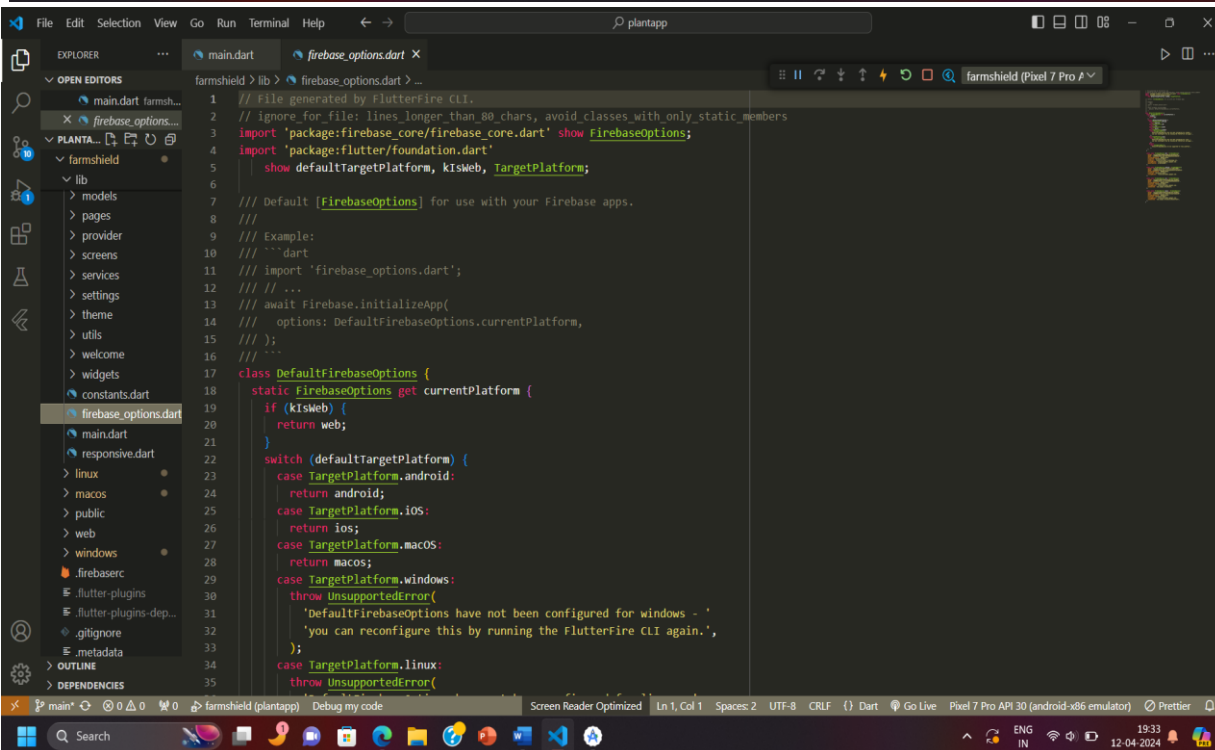
E. Deployment:

The Flutter app can be compiled into native Android and iOS binaries for distribution on app stores. The Python backend can be deployed on a server or cloud platform such as Google Cloud Platform or Amazon Web Services.

Firebase services are integrated into the app during development and can be configured for production deployment.

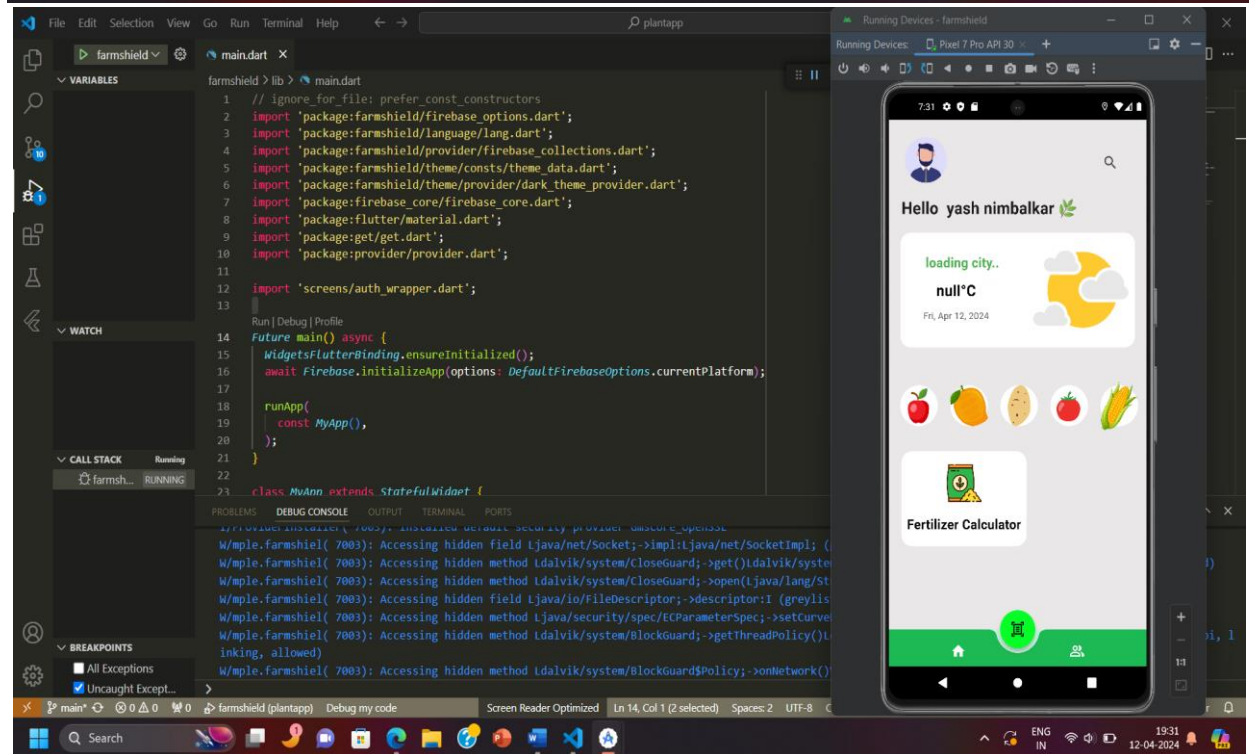
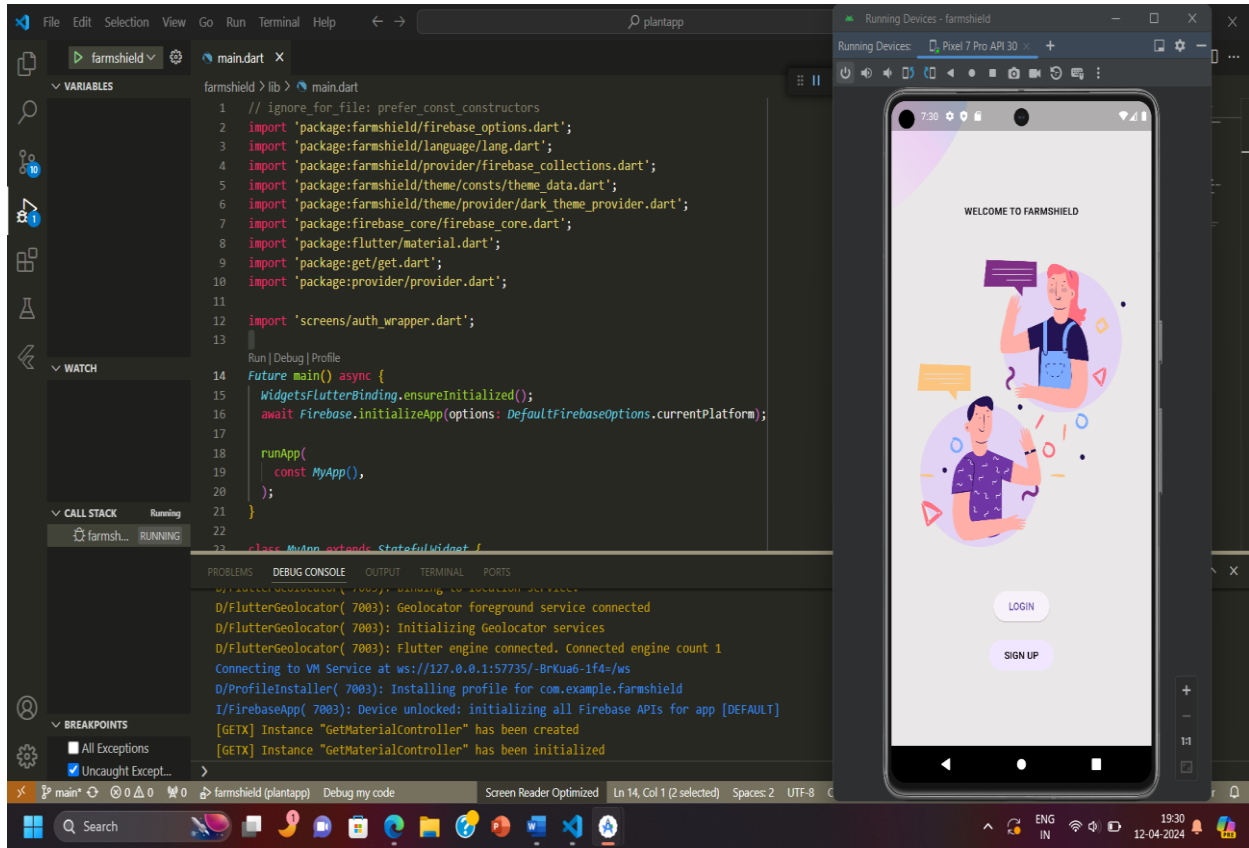


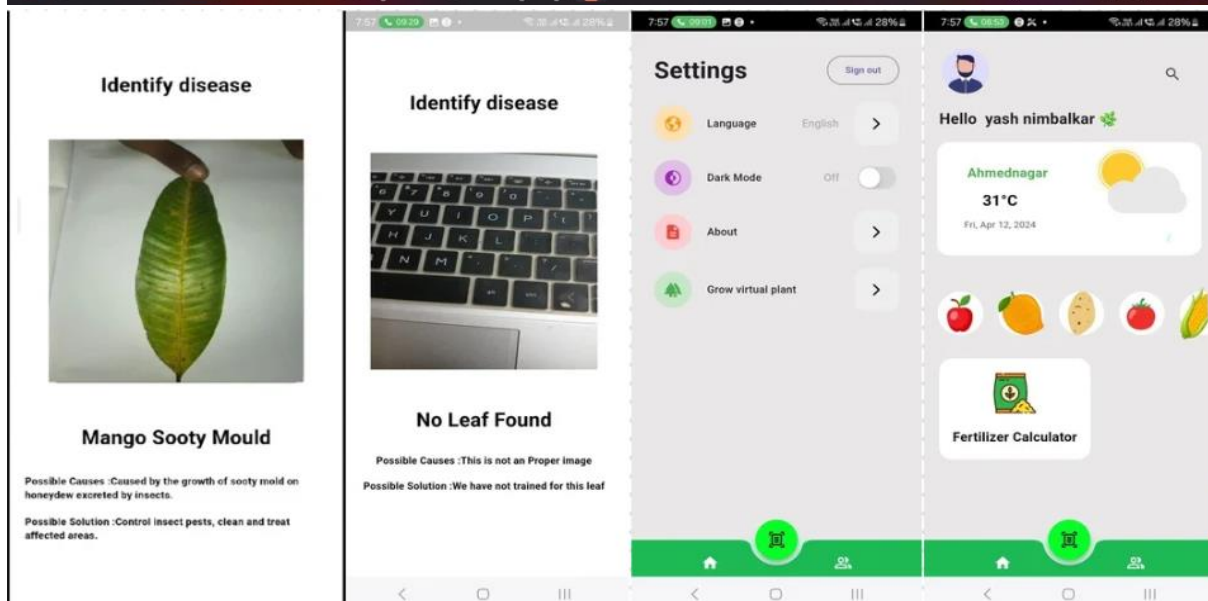
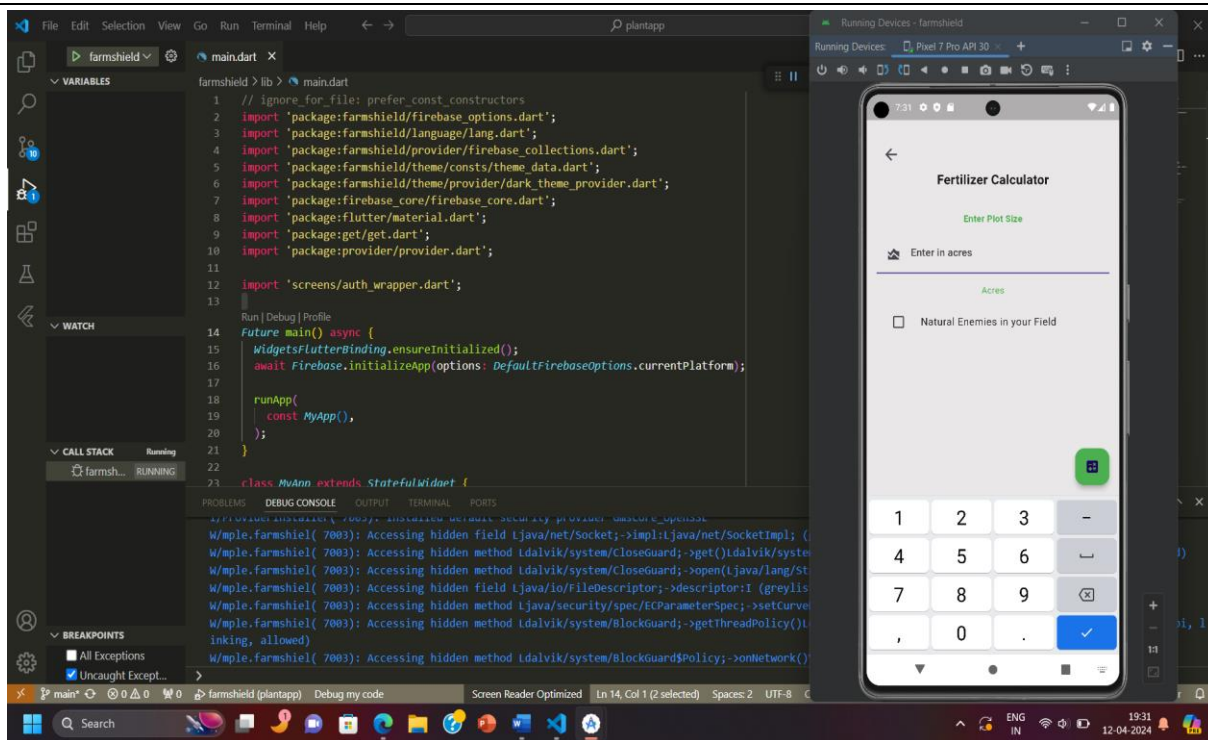
```
1 // File generated by FlutterFire CLI.
2 // ignore_for_file: lines_longer_than_80_chars, avoid_classes_with_only_static_members
3 import 'package:firebase_core/firebase_core.dart' show FirebaseOptions;
4 import 'package:flutter/foundation.dart'
5   show defaultTargetPlatform, kIsWeb, TargetPlatform;
6
7 /// Default [FirebaseOptions] for use with your Firebase apps.
8 ///
9 /// Example:
10 /// ```dart
11 /// import 'firebase_options.dart';
12 /// ...
13 /// await Firebase.initializeApp(
14 ///   options: DefaultFirebaseOptions.currentPlatform,
15 /// );
16 /// ```
17
18 class DefaultFirebaseOptions {
19   static FirebaseOptions get currentPlatform {
20     if (kIsWeb) {
21       return web;
22     }
23     switch (defaultTargetPlatform) {
24       case TargetPlatform.android:
25         return android;
26       case TargetPlatform.iOS:
27         return ios;
28       case TargetPlatform.macOS:
29         return macos;
30       case TargetPlatform.windows:
31         throw UnsupportedError(
32           'DefaultFirebaseOptions have not been configured for windows - '
33           'you can reconfigure this by running the Flutterfire CLI again.',
34         );
35       case TargetPlatform.linux:
36         throw UnsupportedError(
37           'DefaultFirebaseOptions have not been configured for linux - '
38           'you can reconfigure this by running the Flutterfire CLI again.',
39         );
40     }
41   }
42 }
```



```
1 // File generated by FlutterFire CLI.
2 // ignore_for_file: lines_longer_than_80_chars, avoid_classes_with_only_static_members
3 import 'package:firebase_core/firebase_core.dart' show FirebaseOptions;
4 import 'package:flutter/foundation.dart'
5   show defaultTargetPlatform, kIsWeb, TargetPlatform;
6
7 /// Default [FirebaseOptions] for use with your Firebase apps.
8 ///
9 /// Example:
10 /// ```dart
11 /// import 'firebase_options.dart';
12 /// ...
13 /// await Firebase.initializeApp(
14 ///   options: DefaultFirebaseOptions.currentPlatform,
15 /// );
16 /// ```
17
18 class DefaultFirebaseOptions {
19   static FirebaseOptions get currentPlatform {
20     if (kIsWeb) {
21       return web;
22     }
23     switch (defaultTargetPlatform) {
24       case TargetPlatform.android:
25         return android;
26       case TargetPlatform.iOS:
27         return ios;
28       case TargetPlatform.macOS:
29         return macos;
30       case TargetPlatform.windows:
31         throw UnsupportedError(
32           'DefaultFirebaseOptions have not been configured for windows - '
33           'you can reconfigure this by running the Flutterfire CLI again.',
34         );
35       case TargetPlatform.linux:
36         throw UnsupportedError(
37           'DefaultFirebaseOptions have not been configured for linux - '
38           'you can reconfigure this by running the Flutterfire CLI again.',
39         );
40     }
41   }
42 }
```

V. RESULT AND OUTPUT





VI. CONCLUSION

In conclusion, the study of plant disease detection has changed dramatically, incorporating cutting-edge tools such as machine learning, molecular diagnostics, and imaging. Because of the shortcomings of traditional procedures, more precise and efficient ways have been developed. Real-time disease diagnosis is now possible with user-friendly apps because to tools like Flutter, Python, and Firebase. Standardization and scalability are still major obstacles, but cooperation and creativity are viable answers for ensuring world food security.

ACKNOWLEDGEMENTS

We take this opportunity to express my hearty thanks to all those who helped me in the completion of the Project Stage – 2 on “Plant Disease Detection using Deep Learning”.

We would especially like to express my sincere gratitude to **Prof.R.A. Ghadage** my Guide and **Prof. V.V Jagtab** HOD Department of Computer Engineering who extended their moral support, inspiring guidance and encouraging independence throughout this task.

We are also grateful to **Dr. Y. R. Kharde**, Principal of **Shri chhatrapati Shivaji Maharaj College Of Engineering** for his in dispensible support, suggestions

VII. REFERENCES

- [1] Wu, Y. Chen, and J. Meng, "DCGAN-based data augmentation for tomato leaf disease identification," *IEEE Access*, vol. 8, pp. 98716–98728, 2020
- [2] D. Das and C. S. G. Lee, "A two-stage approach to few-shot learning for image recognition," *IEEE Trans. Image Process.*, vol. 29, no. 5, pp. 3336–3350, Dec. 2020
- [3] D. Ashourloo, H. Aghighi, A. A. Matkan, M. R. Mobasheri, and A. M. Rad, "An investigation into machine learning regression techniques for the leaf rust disease detection using hyperspectral measurement," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 9, pp. 4344–4351, Sep. 2016, doi: 10.1109/jstars.2016.2575360
- [4] P. Jiang, Y. Chen, B. Liu, D. He, and C. Liang, "Real-time detection of apple leaf diseases using deep learning approach based on improved convolutional neural networks," *IEEE Access*, vol. 7, pp. 59069–59080, 2019
- [5] B. Sandika, S. Avil, S. Sanat, and P. Srinivasu, "Random forest based classification of diseases in grapes from images captured in uncontrolled environments," in *Proc. IEEE 13th Int. Conf. Signal Process. (ICSP)*, Nov. 2016, pp. 1775–1780.
- [6] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanekharan, "Using deep transfer learning for image-based plant disease identification," *Comput. Electron. Agriculture.*, vol. 173, Jun. 2020, Art. no. 105393.
- [7] S.-N. Ren, Y. Sun, H.-Y. Zhang, and L.-X. Guo, "Plant disease identification for small sample based on one-shot learning," *Jiangsu J. Agricult. Sci.*, vol. 35, no. 5, pp. 1061–1067, May 2019.
- [8] K. Nagasubramanian, S. Jones, A. K. Singh, S. Sarkar, A. Singh, and B. Ganapathysubramanian, "Plant disease identification using explainable 3D deep learning on hyperspectral images," *Plant Methods*, vol. 15, no. 1, pp. 1–10, Aug. 2019
- [9] W. Yang, C. Yang, Z. Hao, C. Xie, and M. Li, "Diagnosis of plant cold damage based on hyperspectral imaging and convolutional neural network," *IEEE Access*, vol. 7, pp. 118239–118248, 2019
- [10] S. P. Mohanty, D. P. Hughes, and M. Salath'e, "Using deep learning for image-based plant disease detection," *Frontiers plant science*, vol. 7, pp. 1419, 2016 IEEE.