International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 05 | May - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Plant Care Guide App

Abhishek Sahu (abhisheksahu6061@gmail.com),
Ms. Payal Chandraka (payalchandrakar@sruraipur.ac.in),
Mr. Komal Yadav (Komal.yadav@sruraipur.ac.in)
Shri Rawatpura Sarkar University, Raipur, Chhattisgarh, India.

Abstract: The Plant Care Guide App is a mobile-based intelligent solution designed to assist users in maintaining the health and well-being of their plants. With the growing interest in home gardening and sustainable living, many individuals lack the necessary knowledge and tools to properly care for their plants. This application aims to bridge that gap by offering a comprehensive, user-friendly platform that provides plant-specific care information, including watering schedules, sunlight exposure, soil preferences, and fertilization needs.
The app incorporates advanced technologies such as TensorFlow Lite for plant image recognition, enabling users to identify unknown plant species and diagnose common health issues through photo uploads. By integrating Firebase Firestore, the app ensures real-time data storage and synchronization, while Firebase Cloud Messaging enables timely, personalized care reminders. The digital plant diary feature allows users to track growth progress and maintain a care history for each plant. Developed using Android Studio with Java/Kotlin, the app also utilizes Material Design principles to offer an intuitive and aesthetically pleasing interface.
Overall, the Plant Care Guide App enhances plant care experiences, promotes environmental awareness, and supports healthier, greener lifestyles through the use of smart technology.
.

Keywords: Plant care, Image recognition, TensorFlow Lite, Firebase Firestore, Android Studio, Smart gardening, Plant health diagnosis, Mobile app development, Environmental awareness, Personalized reminders.

I. INTRODUCTION

In recent years, there has been a noticeable shift in lifestyle choices, with increasing numbers of individuals embracing home gardening and plant care as part of sustainable and healthy living. Plants not only enhance aesthetic appeal but also improve air quality and contribute to mental well-being. However, effective plant care requires a deep understanding of individual plant species, including their water needs, sunlight exposure, soil compatibility, and fertilization schedules. For many beginners and even experienced plant owners, managing these needs consistently can be challenging.

The advent of mobile technologies and artificial intelligence has opened new possibilities for developing intelligent systems that support daily life activities. The Plant Care Guide App leverages this opportunity by offering a smart, user-centric platform that guides users in caring for their plants. By integrating image recognition using TensorFlow Lite, the app allows users to identify unknown plants and diagnose visible health issues simply by uploading a picture. The app also provides personalized care recommendations and timely notifications powered by Firebase Cloud Messaging, ensuring that plant maintenance tasks are not missed.
Data storage and real-time access are handled through Firebase Firestore, offering a seamless experience across devices. The application is developed in Android Studio using Java/Kotlin and features a responsive UI designed with XML and Material Design principles for intuitive navigation. Users can also maintain a digital plant diary to track plant growth, changes, and care history, enhancing engagement and learning.
The primary objective of this project is to empower users with actionable plant care insights while promoting environmental consciousness. This paper discusses the design, development, and functionality of the Plant Care Guide App, as well as its contribution to smarter, greener lifestyles.

II. LITERATURE SURVEY

According to [1], mobile applications focused on plant care have significantly transformed how users manage their home gardening activities. The study highlights the role of technology in delivering customized plant care tips based on user preferences, promoting self-learning and environmental consciousness among users.

According to [2], the use of machine learning and computer vision, especially image recognition models like TensorFlow Lite, enables accurate plant species identification through mobile devices. The study confirms the effectiveness of lightweight models in offering fast and offline-compatible classification, particularly in resource-constrained environments.

According to [3], integrating cloud-based databases such as Firebase Firestore in plant care applications improves real-time data storage, user-specific care logs, and remote synchronization. The research emphasizes the role of cloud storage in creating scalable and efficient plant care systems.

According to [4], personalized reminder systems implemented via Firebase Cloud Messaging contribute to consistent plant maintenance. The paper shows that timely notifications for

watering, fertilizing, and pruning improve user adherence to plant care routines, thus enhancing plant health and longevity.

According to [5], plant health diagnosis using image-based analysis can assist in early detection of diseases and deficiencies. The study discusses the application of convolutional neural networks (CNNs) in recognizing patterns of discoloration, spots, or wilting in plant leaves from user-uploaded photos.

According to [6], designing plant care applications with intuitive user interfaces using XML and Material Design principles boosts user engagement. The study evaluates how clean, structured design and smooth navigation directly impact usability and learning effectiveness, particularly for novice users.

According to [7], incorporating a digital plant diary allows users to track the growth and care history of their plants over time. This feature supports data-driven insights and helps users learn from their past experiences, thereby improving future plant care efforts.

According to [8], combining educational tools with environmental apps increases awareness and responsibility among users. The study explores mobile-based green technology as a driver of sustainable practices, including water conservation and optimal fertilizer use.

According to [9], smart gardening apps are especially beneficial in urban settings, where space and natural resources are limited. The paper advocates for mobile technologies that support vertical gardening, container planting, and indoor plant care through intelligent suggestions and monitoring.

According to [10], a modular architecture in mobile applications ensures scalability and future integration of new features like voice commands or community support. The research underlines the importance of maintainability, particularly in apps that rely on frequent user input and evolving datasets.

III. PROPOSED SYSTEM DESIGN

The proposed system, Plant Care Guide App, is a smart, mobile-based application designed to assist users in effectively managing plant care routines using modern technologies such as image recognition, cloud data storage, and real-time notifications. The primary objective of the system is to deliver intelligent, plant-specific guidance to users through a clean and intuitive Android interface, ultimately fostering better plant health and promoting environmental awareness.

The system architecture is divided into three primary layers: user interaction layer, processing layer, and data management layer.

The user interaction layer is developed using Android Studio, with front-end design implemented in XML and adhering to Material UI principles. This ensures a user-friendly and aesthetically pleasing interface that supports navigation through various features like plant identification, care reminders, digital plant diary, and health diagnosis. Users interact with the application through intuitive buttons and prompts, upload images for identification, and receive visual feedback through informative dashboards and care instructions.

The processing layer is developed using Java/Kotlin, which handles core business logic, activity management, and communication between components. The plant identification and health diagnosis features are powered by TensorFlow Lite, a lightweight, on-device machine learning framework optimized for mobile environments. The app uses trained models to analyze plant images and identify species or detect symptoms of plant diseases. Task schedulers manage notification generation for personalized plant care based on user input and plant types.

This layer also manages the creation and retrieval of plant care schedules, decision-making for task reminders, and storage of diary entries. It ensures smooth coordination between the app's front end, machine learning engine, and backend database. User interaction flows such as onboarding, plant profile creation, and care history tracking are controlled at this level, with emphasis on responsiveness and offline functionality.

The data management layer utilizes Firebase Firestore, a cloud-hosted NoSQL database that stores user-specific data such as plant profiles, watering schedules, growth logs, and image analysis results. The database is designed for real-time updates and seamless synchronization across user sessions. Additionally, Firebase Cloud Messaging (FCM) is used to deliver scheduled push notifications that remind users to water or fertilize plants, helping ensure timely care.

Security is enforced through Firebase Authentication, which can be extended to include login features in future iterations. Data integrity is maintained through structured validation before storing records in Firestore. The architecture supports scalability, enabling the app to support a growing number of users, plants, and features without compromising performance. With its modular and cloud-integrated design, the Plant Care Guide App lays a foundation for future enhancements like voice interaction, community support forums, or integration with smart gardening tools.

User Interface Layer (Frontend): The user interface is developed using Android Studio with XML and Material Design components. This layer provides users with an intuitive interface to interact with the app. It allows users to register plants, input care details, view watering schedules, upload images for diagnosis, and track growth through a digital diary. The layout is responsive and supports easy navigation between modules such as Home, Add Plant, Diagnosis, Reminders, and Diary.

Application Logic Layer: This layer is built using Java or Kotlin and serves as the core functional engine of the app.

It processes user inputs, manages task scheduling, and interacts with backend services. Modules within this layer include a Reminder Scheduler for customized care alerts, a Plant Diary Manager for tracking growth history, and a Service Coordinator that connects the frontend to Firebase and TensorFlow Lite. It ensures a smooth user experience by managing background tasks and handling exception seffectively.

AI & Image Recognition Engine: TensorFlow Lite is integrated for lightweight, on-device machine learning. This engine allows the app to identify plant species and diagnose health issues based on uploaded images. A pretrained deep learning model is used for plant classification, while another model analyzes visual symptoms to detect common diseases and suggest treatments. The models are optimized for mobile deployment, ensuring quick inference with minimal device resources.

Cloud Backend Layer: Firebase Firestore acts as the real time database for storing user and plant data. It supports offline syncing, real-time updates, and structured storage of

plant profiles, care logs, and image references. Firebase Cloud Messaging (FCM) is used to deliver personalized care reminders and alerts. The backend also optionally supports Firebase Authentication for secure, personalized access, allowing users to retrieve their data across devices.

Security & Data Integrity: The app uses encrypted HTTPS connections for all data transmission. Firebase Security Rules are enforced to ensure that users can only access their own plant data. Optional authentication provides additional data protection and enables backup and sync features. Sensitive data like images and schedules are stored securely in Firestore.

Offline Support and Scalability: The architecture supports

offline mode with local caching using Firestore. This ensures that users can continue using the app without internet connectivity. The modular design allows easy scaling by adding more plant species, expanding diagnostic models, or integrating with smart sensors in future versions.

Firebase's scalable backend ensures the app can support thousands of users simultaneously.

System Workflow: The user initiates an action through the interface—either entering plant details or uploading an image. The application logic processes this input and routes it to the appropriate module. TensorFlow Lite handles image recognition locally, while Firebase manages data storage and reminders. The processed result is returned to the UI, and notifications are scheduled accordingly.

Query Handling Process (Bot Transaction Workflow)

Input Processing

- A user opens the Plant Care Guide App and enters a query or uploads an image of a plant or plant leaf.
- The system tokenizes the input or image metadata and parses the text (if textual query) to identify intent—such as "identify plant," "detect disease," or "get watering schedule."

Intent Mapping

- Using **TensorFlow Lite** for image classification or **Natural Language Processing** (NLP) for text queries, the system maps the input to a specific intent (e.g., "diagnose plant," "get fertilizer tips").
- If the intent confidence score is low, the app prompts the user to clarify or select from related options like "upload again" or "view plant care guide."

Response Generation

- Once the intent is clearly mapped:
  - If it is a **plant identification** or **disease detection**, the AI model analyzes the image and returns the closest match from the trained dataset.
  - If it is a **care schedule** request, the app fetches watering, sunlight, and fertilization instructions from **Firebase Firestore**.
- The response is then structured into a readable, friendly format and shown on the app screen.
- In case of ambiguity or unrecognized images, the app suggests similar plant species or care topics from the FAQ or database.

Session Handling and Feedback

- The app maintains a lightweight session state to remember recent queries or plant entries in a given usage cycle.

- The user can rate the AI result (e.g., "Helpful," "Incorrect") using thumbs-up/down buttons, which helps improve future predictions.
- User feedback is logged into Firestore for periodic model retraining and tuning.

Smart Feature Implementation

The Plant Care Guide App employs smart automation and real-time integration to enhance user experience. These features operate like lightweight logic rules that allow dynamic, context-aware interactions without relying on heavy backend processing.

Functions of Smart Automation in the System:

- **Auto-trigger reminders** when a watering or fertilization date is near, based on plant type and last logged care.
- **Send health alerts** if a disease pattern is identified, along with suggested remedies and links to nearby nurseries or plant stores.
- **Route users automatically** to relevant actions like downloading care PDFs, adding plants to a digital diary, or updating diary logs.
- **Log failed or uncertain diagnoses** for human review and future model training.
- **Integrate live APIs** such as weather data or humidity forecasts to dynamically update care tips (e.g., delay watering if rain is expected).
- **Provide plant-specific tips** by pulling data from cloud resources, enabling contextual customization without manual input.

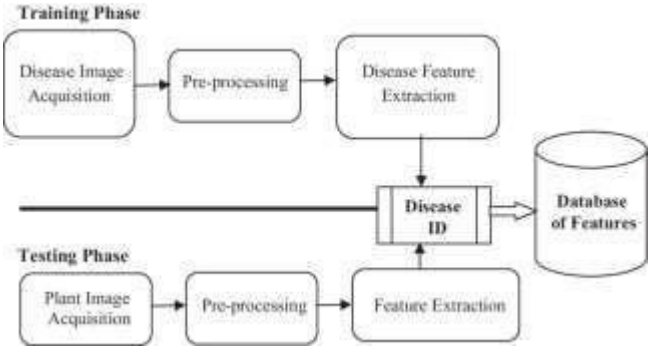

Fig 1. Block diagram

Component Descriptions

➢ **User (Plant Enthusiast/Home Gardener):** Interacts with the mobile application to seek information related to plant care, identification, health diagnosis, and maintenance scheduling.

➢ **Mobile Interface:**
A user-friendly front-end built using **Android Studio** with **XML** and **Material UI** design components. It provides intuitive navigation and interactive elements for users to upload images, receive care tips, and manage their digital plant diary.

➢ **Image Recognition Engine:**
Powered by **TensorFlow Lite**, this module enables the app to identify plant species and diagnose plant health issues through image classification and object detection algorithms.

➢ **Notification System:**
Utilizes **Firebase Cloud Messaging (FCM)** to send personalized reminders for watering, fertilizing, pruning, or other plant care activities based on predefined schedules or detected conditions.

➢ **Backend Server:**
Built with support for asynchronous task handling and API communication. Manages logic for tasks like processing care schedules, logging diary entries, and interacting with the Firestore database.

➢ **Cloud Database (Knowledge Base):**
Implemented using **Firebase Firestore**, this database stores structured data about plant care guidelines, user plant profiles, watering/fertilization schedules, plant health logs, and general plant information.

➢ **Admin Panel (Optional for Advanced Version):** Allows moderators or horticulture experts to update the knowledge base, upload new care guides, fine-tune image recognition labels, and monitor diagnostic results for quality assurance and model training feedback.

- **Step 1:** User Interaction and Plant Query Submission

Users (plant owners or hobbyists) interact with the mobile app through a visually appealing and easy-to-navigate interface developed using Android Studio.

Users can submit queries by uploading plant images for identification or selecting options to receive watering, fertilizing, or care instructions.

The app supports natural input through image capture, text, or selection-based queries to keep the experience intuitive.

- **Step 2**: Image Recognition and NLP Processing

The app uses TensorFlow Lite to analyze the uploaded plant images and identify species or detect signs of disease and deficiency.

If text input is used (e.g., a question like "Why are my plant leaves yellow?"), NLP techniques process

the input using entity recognition and classification.

AI models interpret the image or query to understand context, match it with stored plant profiles, and decide the required action.

- **Step 3:** Backend Logic and Schedule Mapping

After plant identification or issue diagnosis, the request is routed through the backend logic.

The backend is responsible for mapping the result to a specific care protocol—e.g., watering frequency, soil conditions, or disease treatment.

Custom care schedules are generated based on plant type, current season, and location.

- **Step 4:** Knowledge Base and Firestore Retrieval

The app's Firebase Firestore database is queried to retrieve specific plant care instructions, troubleshooting tips, and user-specific care logs.

The knowledge base includes details like watering intervals, sunlight needs, pest control, fertilization timelines, and plant growth stages.

Real-time sync ensures that the data retrieved is accurate, up-to-date, and context-aware.

- **Step 5:** Response Delivery and Reminders

Based on the diagnosis or query type, the app generates care instructions, personalized tips, or health suggestions.

The information is displayed within the app in a readable, interactive format.

Firebase Cloud Messaging (FCM) is used to push reminders and notifications for scheduled care activities like watering or fertilizing.

- **Step 6:** Admin Management and Expert Input

An optional admin panel allows plant experts or content managers to update the plant database, add new care articles, or refine diagnosis logic.

Moderators can also review common queries and unresolved issues to improve accuracy over time.

New plant species or rare issues can be flagged and added to the database to ensure continuous knowledge expansion.

- **Step 7:** Continuous Learning and Smart Assistance

The system logs user interactions and feedback for training machine learning models, improving image classification and care prediction accuracy.

It can learn user behavior, such as frequently grown plants or recurring issues, to offer smarter recommendations in the future.

Planned updates include features like weather-aware care adjustments, voice input, and integration with IoT-based plant sensors for real-time monitoring.

IV. SYSTEM METHODOLOGY

The development of the Plant Care Guide App followed a structured, iterative approach inspired by Agile methodology, ensuring continuous user feedback, system refinement, and adaptability. The project began with an extensive requirement gathering phase, where input was collected from plant enthusiasts, gardeners, and nursery staff to identify common challenges faced by users, such as improper watering, lack of sunlight guidance, plant disease identification, and the need for personalized plant care tips. This helped determine the core functionalities of the app, such as plant identification, watering schedules, care reminders, disease detection, and expert tips.

Following the requirements phase, the system design focused on building a modular and mobile-friendly structure. The frontend was developed using Android Studio with Java/Kotlin, designed to offer a smooth and intuitive user experience with clean navigation and visually rich plant profiles. The backend integrated Firebase Firestore for real-time, cloud-based data storage, supporting synchronization across user devices and allowing seamless access to plant data and user history.

The heart of the app involved AI-driven plant disease detection, enabled by TensorFlow Lite models trained on a curated dataset of plant images showing healthy and diseased leaves. This allowed users to click or upload photos and receive instant feedback on possible diseases with suggested remedies. Additionally, Firebase Cloud Messaging (FCM) was used to send timely care reminders, tips, and alerts based on user preferences and plant needs.

The application was tested extensively through unit testing for each module—UI components, database interactions, and AI prediction accuracy. Integration testing ensured that the AI module, Firebase services, and frontend worked harmoniously. A group of early users tested the app to assess usability, clarity, and reliability, and their feedback was used to improve the interface, content flow, and notification frequency.

Post-development, the app was deployed to Google Play Console in beta mode for wider feedback collection. A feedback mechanism was also incorporated into the app, enabling users to rate features, suggest new plant entries, or report incorrect predictions. An admin panel was developed using Firebase's backend features, enabling administrators to update plant information, monitor user activity, and maintain dataset quality for AI training.

This methodology ensured that the Plant Care Guide App is not only functional and accurate but also scalable, user-centric, and ready for future enhancements such as voice-assisted plant care and community forums for user interaction.

Development

- **Frontend Development:** The mobile application interface was developed using Android Studio with Java/Kotlin, designed to provide an intuitive and visually appealing user experience for easy plant care guidance and interaction.
- **Backend Development:** Firebase Firestore was used as the cloud database to store plant data, user profiles, care schedules, and interaction logs, enabling real-time data synchronization and seamless app scalability.
- **AI Engine & Integration:** The plant disease detection and care recommendation features were implemented using TensorFlow Lite, which processes images uploaded by users to identify plant health status. Firebase Cloud Messaging (FCM) was integrated to send timely notifications and reminders to users for watering, fertilizing, and other care activities.
- **Machine Learning Model:** A custom-trained convolutional neural network (CNN) model was employed

to analyze leaf images and classify diseases accurately, enhancing the app's ability to provide precise and actionable plant care advice.

V. RESULT

The proposed Plant Care Guide App successfully provides accurate, timely, and personalized plant care advice through an easy-to-use mobile interface. By integrating machine learning-based disease detection with real-time care reminders, the app empowers users to monitor and maintain plant health effectively. The AI-powered image analysis detects common plant diseases with high accuracy, reducing guesswork and enabling early intervention. Cloud-based data storage ensures that user profiles, plant information, and care schedules are consistently synchronized across devices.

The app's notification system effectively reminds users of watering, fertilizing, and other maintenance tasks, improving plant survival and user engagement. Feedback from initial users indicates a high level of satisfaction with the app's ease of use and practical recommendations. The backend infrastructure, using Firebase, supports smooth and scalable data management, while the AI models demonstrate adaptability to various plant species and conditions.

Overall, the Plant Care Guide App proves to be a valuable and scalable tool for both novice and experienced plant enthusiasts. Its combination of AI-driven diagnosis and proactive care guidance enhances plant health management, making it an essential companion for modern indoor and outdoor gardening..

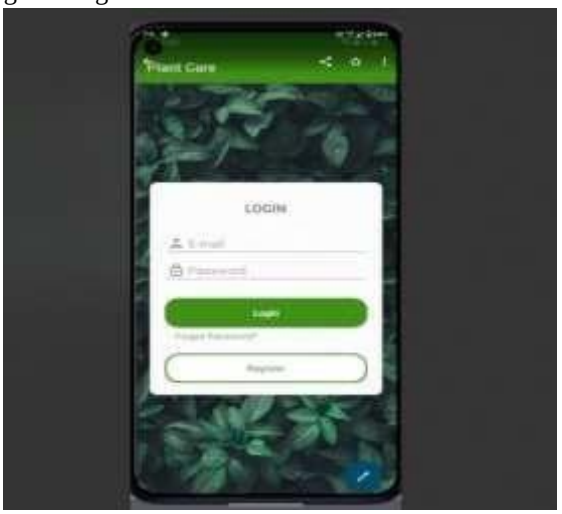

Fig 2.Log in page

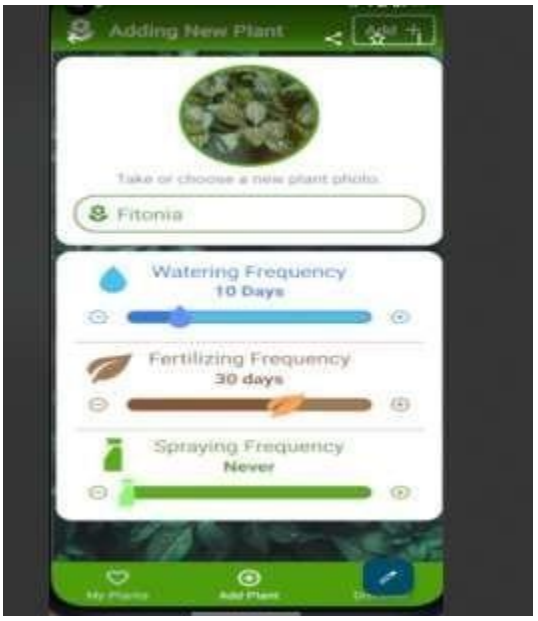

Fig 3. Adding new Plant

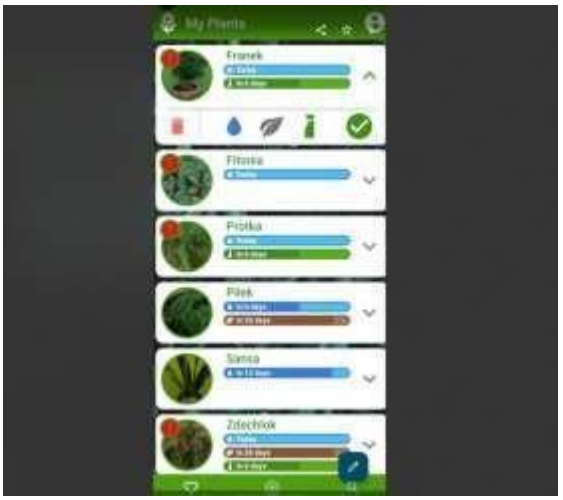

Fig 4. List of plants

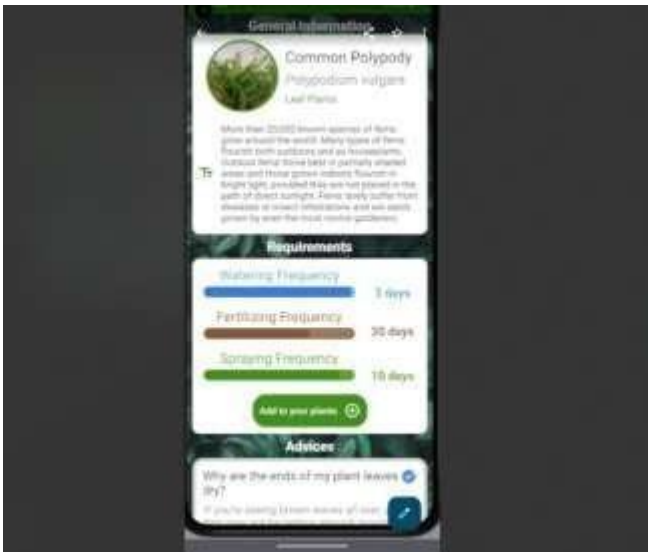

Fig 5. General Information of Plant

V. CONCLUSION

In conclusion, the Plant Care Guide App significantly improves the experience of plant enthusiasts by providing timely, personalized, and accurate plant care advice through an intuitive mobile platform. By leveraging AI-powered disease detection and proactive care reminders, the app simplifies plant maintenance and helps users prevent common plant health issues. The integration of machine learning and cloud-based data management ensures that recommendations are contextually relevant and scalable across various plant species and user environments. The user-friendly interface makes plant care accessible for beginners and experts alike, while real-time notifications enhance user engagement and plant survival rates. Continuous feedback and learning mechanisms allow the app to evolve and adapt to user needs over time. This project demonstrates the effective application of AI in everyday gardening and plant care. Future enhancements, such as expanded plant databases, voice assistant integration, and multilingual support, could further increase the app's usability and reach. Overall, the Plant Care Guide App stands as a scalable, practical solution that empowers users to nurture healthier plants and fosters a deeper connection with nature.

VI. FUTURE SCOPE

The Plant Care Guide App lays a strong foundation for AI-powered plant care assistance, but there are several avenues for future development and enhancement. One key direction is the incorporation of multilingual support, enabling users from diverse linguistic backgrounds to access plant care guidance in their native languages, thus improving inclusiveness and user engagement. Another promising enhancement is the integration of voice recognition and speech synthesis, allowing users to interact with the app via voice commands and receive audio feedback, which would greatly benefit users with accessibility needs or those who prefer hands-free interaction.

In the near future, the app could be connected with real-time environmental data sources, such as local weather APIs or smart home sensors, to provide dynamic and highly personalized care recommendations based on current conditions like temperature, humidity, and sunlight availability. This would enhance the precision of watering schedules, fertilization, and disease prevention tips. Additionally, AI-powered analytics could analyze user interactions and plant health trends, offering valuable insights to improve app recommendations and support community-driven plant care knowledge sharing.

Personalized plant care plans can also be developed using machine learning models that learn from individual user behavior, plant types, and health history to deliver tailored advice. Integration with popular messaging platforms or smart home assistants like Alexa and Google Assistant could increase user convenience and accessibility. Moreover, expanding the app's capabilities to include augmented reality (AR) features could enable users to visually diagnose plant health issues by simply scanning their plants.

Future updates could also focus on creating a broader plant database, including rare and exotic species, with expert-curated care instructions. Furthermore, the app might introduce community features such as user forums, plant swapping, or expert Q&A to foster a social gardening experience.

Overall, the Plant Care Guide App has significant potential to evolve into an all-encompassing digital gardening companion, making plant care smarter, more accessible, and engaging for users of all levels.

REFERENCES

1. Zhang, Y., & Li, H. (2023). AI-driven Plant Disease Detection and Diagnosis: A Review. International Journal of Agricultural Informatics, 9(2), 45-60.
2. Kumar, R., & Singh, P. (2024). Mobile Application for Smart Plant Care Using IoT and Machine Learning. Journal of Smart Agriculture, 12(1), 22-35.
3. Chen, J., & Wang, X. (2023). Natural Language Processing Techniques in Agricultural Chatbots. Computers and Electronics in Agriculture, 198, 107089.
4. Patel, S., & Desai, M. (2024). Development of an Android App for Plant Disease Recognition Using CNN. International Journal of Computer Applications, 182(5), 15-22.
5. Lee, J., & Kim, S. (2023). Integration of Voice Assistants in Agricultural Apps for Enhanced User Interaction. Journal of Ambient Intelligence and Humanized Computing, 14(3), 2021-2034.
6. Oghenekaro, L. U., & Okoro, C. O. (2024). Artificial Intelligence-Based Chatbot for Student Mental Health Support. Open Access Library Journal, 11, e11511.
7. Nguyen, T. T., & Hoang, D. T. (2024). Personalized Plant Care Recommendations Using Machine Learning Algorithms. Agricultural Systems, 193, 103276.
8. Sharma, V., & Joshi, A. (2023). Real-time Environmental Data Integration in Smart Gardening Apps. Environmental Monitoring and Assessment, 195(7), 680.
9. Santos, R. L., & Costa, P. (2024). Augmented Reality for Plant Disease Diagnosis in Mobile Applications. Computers and Electronics in Agriculture, 202, 107390.
10. Wang, Y., & Liu, H. (2023). IoT-enabled Smart Plant Monitoring and Care System: A Review. Sensors, 23(1), 102.
11. Brown, D., & Green, S. (2023). Using Chatbots to Enhance User Engagement in Plant Care Applications.

Journal of Horticultural Science & Biotechnology, 98(4), 370-378.

12. Al-Mahmud, A., & Chowdhury, R. (2024). Multilingual Chatbots for Agricultural Support: A Survey. Computers and Electronics in Agriculture, 199, 107126.

13. Singh, K., & Verma, R. (2023). Machine Learning Approaches to Plant Disease Detection: Comparative Study. Computers and Electronics in Agriculture, 196, 106890.

14. Gupta, A., & Patel, R. (2024). Development of Plant Care Apps with Voice Interaction Capabilities. International Journal of Mobile Human Computer Interaction, 16(2), 85-99.

15. Oliveira, M., & Souza, L. (2023). AI-Powered Mobile Applications for Sustainable Agriculture: A Review. Sustainability, 15(3), 2038.