AGROAID: EMPOWERING SMALL-SCALE FARMERS WITH MULTILINGUAL AI ASSISTANCE

PROJECT REPORT

Submitted by

BHARATH KUMAR N	**727621BIT064**
NAVEEN V C	**727621BIT074**
DINESH M	**727622BIT320**

in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

INFORMATION TECHNOLOGY

DR. MAHALINGAM COLLEGE OF ENGINEERING AND TECHNOLOGY

AN AUTONOMOUS INSTITUTION

AFFILIATED TO ANNA UNIVERSITY

CHENNAI 600 025

APRIL 2025

DR. MAHALINGAM COLLEGE OF ENGINEERING AND TECHNOLOGY, POLLACHI -642 003

(AN AUTONOMOUS INSTITUTION AFFILIATED TO ANNA UNIVERSITY, CHENNAI - 600 025)

BONAFIDE CERTIFICATE

Certified that this project report titled "AgroAid: Empowering Small-Scale Farmers with Multilingual AI Assistance" is

the bonafide work of

STUDENT NAME	ROLL NO
BHARATH KUMAR N	**727621BIT064**
NAVEEN V C	**727621BIT074**
DINESH M	**727622BIT320**

who carried out the project work under my supervision.

Dr. L. MEENACHI M.E., Ph.D.,

ASSOCIATE PROFESSOR and HOD i/c
Department of IT
Dr. Mahalingam college of Engineering and Technology
Pollachi- 642003

Dhiyaneswaran.J,

SUPERVISOR
Assistant Professor (SS)
Department of IT
Dr. Mahalingam college of Engineering and Technology
Pollachi- 642003

Submitted for the Autonomous End Semester project Viva-Voce Examination held on _____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

Dr. Mahalingam College of Engineering and Technology, Pollachi-642003

Academic Year: 2024 - 2025

TRL, SDG and Similarity Compliance Certificte

Project Title: AgroAid: Empowering Small-Scale Farmers with Multilingual AI Assistance**.**

Course Code: 19ITPN6801

Department and Semester: Information technology / 8th

Technology Readiness Level (TRL) of the Project : _____

Sustainability Development Goals (SDG)-Goal Name : _____

Similarity % from Turnitin Software : _____

I/we hereby declare that this project report is original and complies with the institution's similarity guidelines.

S.No.	Names and Roll Numbers of Students	Signature of the Students
01	BHARATH KUMAR N – 727621BIT064	
02	NAVEEN V C -727621BIT074	
03	DINESH M-727622BIT320	

Verified by

Name & Signature of the Guide **Head of the Department**

Endorsed by

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

ABSTRACT

The core architecture of agroaid is built using a layered approach, integrating data acquisition, processing, model inference, and user interaction. Machine learning models are trained using publicly available agricultural datasets, such as those from the indian council of agricultural research (icar) and international sources like fao and kaggle. Feature engineering plays a crucial role in improving model accuracy, with inputs such as soil ph, nutrient levels, temperature, humidity, and historical crop performance being processed to derive actionable insights. The image recognition module for pest and disease detection utilizes convolutional neural networks (cnns), fine-tuned on region-specific agricultural image datasets to ensure high accuracy and relevance.

Agroaid has the potential to significantly improve yield and resource efficiency for smallholder farmers, particularly in regions where access to agronomic expertise is limited. By providing localized and timely advice, the system helps mitigate risks associated with climate change, soil degradation, and pest outbreaks. Furthermore, the multilingual and voice-enabled interface bridges literacy gaps, making advanced agri-tech accessible to farmers in rural and semi-urban areas. The inclusion of government scheme recommendations ensures that users are not only optimizing agricultural practices but also leveraging institutional support, thereby enhancing economic resilience.

Looking forward, the agroaid platform is designed with scalability in mind. Future iterations may include iot-based sensor integration for real-time field monitoring, enabling dynamic recommendations based on live soil moisture, temperature, and weather data. A mobile application version is also planned to increase accessibility and convenience for farmers on the move. In addition, partnerships with local agricultural extension centers and ngos can aid in widespread adoption and training. By continuously incorporating feedback from end-users and stakeholders, agroaid aims to evolve into a comprehensive digital farming assistant, supporting the global push towards smart, sustainable, and inclusive agriculture.

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

We wish to express our sincere thanks to all who have contributed to do this project through their support, encouragement and guidance.

We extend our gratitude to our management for having provided us with all facilities to build my project successfully. We express our sincere thanks to our honourable Secretary, Dr.C.Ramaswamy, M.E., Ph.D., F.I.V., for providing the required amenities.

We take this opportunity to express our deepest gratitude to our principal Dr.P.Govindasamy, Ph.D., who provide suitable environment to carry out the project.

We heartily express our extreme gratefulness to Dr. S. Ramakrishnan, M.E., Ph.D., Professor-IT & Dean–RI for his constant motivation and wonderful support to us.

We extend our heartfelt gratitude to Dr.L.MEENACHI, M.E., Ph.D., Professor and HODi/c of the Department of Information Technology, for her tremendous support and encouragement.

We wish to express our deep sense of gratitude and thankfulness to my project guide Mr.Dhiyaneeswaran J, for the valuable suggestion and guidance offered during the course of the project.

Finally, we are committed to place our heartfelt thanks to all those who had contributed directly and indirectly towards the success of the completion of this project.

LIST OF CONTENTS

TABLE OF CONTENTS

LIST OF TABLES

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES

LIST OF ABBREVIATIONS

LIST OF ABBREVIATIONS

1. AI – ARTIFICIAL INTELLIGENCE
2. NLP – NATURAL LANGUAGE PROCESSING
3. CV – COMPUTER VISION
4. ML – MACHINE LEARNING
5. IOT – INTERNET OF THINGS
6. TTS – TEXT-TO-SPEECH
7. GUI – GRAPHICAL USER INTERFACE
8. API – APPLICATION PROGRAMMING INTERFACE
9. IASS – INTEGRATED AGRICULTURAL SUPPORT SYSTEM
10. FARMS – FERTILIZER AND AGRICULTURAL RECOMMENDATION MANAGEMENT SYSTEM

CHAPTER 1

INTRODUCTION

CHAPTER 1

1.1 INTRODUCTION

Agriculture forms the backbone of rural economies in countries like India, where a significant proportion of the population depends on farming for their livelihood. Despite its vital role, small-scale farmers continue to face challenges such as limited access to expert guidance, inadequate knowledge of modern techniques, and difficulties in navigating government schemes. These issues are further compounded by language barriers, lack of real-time data, and insufficient technological infrastructure in remote areas.

In recent years, artificial intelligence (AI) and machine learning (ML) have shown remarkable potential in transforming traditional farming practices. By leveraging historical data, predictive modeling, and intelligent automation, AI can support farmers in making better-informed decisions regarding crop selection, soil management, irrigation, and pest control. AgroAid is a step forward in this direction—a smart, AI-powered agricultural assistant designed to bridge the gap between farmers and technology.

This project introduces AgroAid, a comprehensive chatbot system developed with a strong focus on multilingual support, usability, and domain relevance. AgroAid empowers small-scale farmers by offering personalized recommendations for crops and fertilizers based on soil type, detecting plant diseases using computer vision techniques, and optimizing irrigation schedules using real-time weather and soil data. Additionally, it integrates information from local and national government databases to inform users about relevant subsidies, financial aid, and farming schemes—often overlooked due to complex application processes and language limitations.

What sets AgroAid apart is its bilingual interface, enabling interaction in both English and Tamil, making the system more accessible to local farmers. It also incorporates text-to-speech (TTS) functionality for voice-based guidance and features a Tkinter-based GUI to cater to users with minimal technical literacy. The modular design of the platform ensures scalability, allowing future integration of advanced features such as weather alerts, market price predictions, and mobile application support.

In essence, AgroAid combines the power of AI, NLP, IoT, and computer vision to deliver a practical, intelligent, and inclusive solution for farmers.

1.2 PROBLEM DEFINITION

Agriculture is the backbone of rural economies, yet small-scale farmers often face challenges such as limited access to expert advice, inefficient resource use, and lack of awareness about government schemes. These issues are worsened by language barriers and low digital literacy, making it difficult for farmers to adopt modern, data-driven farming practices.

The core problem this project addresses is the absence of an intelligent, accessible system that provides personalized agricultural insights in the farmer's native language. Although agricultural data exists, it is rarely transformed into actionable guidance tailored to the local context of small-scale farmers.

- **Crop and fertilizer recommendation:** farmers need reliable support to select the right crops and fertilizers based on soil type, nutrient levels, and climate conditions.
- **Pest and disease detection:** identifying plant diseases manually can lead to delays and crop loss. An ai-powered visual detection system can provide real-time support.
- **Government scheme awareness:** many farmers are unaware of beneficial subsidies and schemes due to complex processes and language limitations.

This project aims to bridge the gap between raw agricultural data and field-level intelligence by combining machine learning, computer vision, and nlp. It offers:

- A multilingual chatbot that delivers crop, fertilizer, and irrigation advice.
- Real-time disease detection through image processing.
- Simplified access to government schemes tailored to user profiles.

Agroaid presents a data-driven solution that supports small-scale farmers in making informed decisions, improving productivity, and ensuring sustainable, inclusive agricultural development.

1.3 PROJECT OVERVIEW

The AgroAid smart farming assistant is a multilingual, ai-driven chatbot system designed to support small-scale farmers by offering personalized agricultural recommendations. The system utilizes machine learning, computer vision, and natural language processing to deliver real-time insights across key areas such as soil analysis, pest detection, irrigation planning, and access to government schemes.

Data integration & processing: the system leverages curated datasets, including soil health records, crop requirements, and pest symptom databases. Categorical inputs such as soil type and crop name are encoded and processed for personalized recommendation generation.

Machine learning-based recommendations: AgroAid uses trained models to predict suitable crops and fertilizers based on soil characteristics. It also incorporates ai-driven logic for irrigation and harvest timing, ensuring efficient resource utilization and improved yield.

Disease detection via image analysis: using computer vision, AgroAid detects common crop diseases by analysing leaf images. Farmers receive instant diagnoses and suggested treatments, reducing the time between detection and action.

Multilingual interface development: the application offers two user-friendly interfaces:

- **Tkinter Desktop GUI:** a lightweight interface with both English and Tamil support, designed for offline use in rural areas.
- **Voice And Text Interaction:** powered by nlp and text-to-speech, enabling farmers to interact with the bot in their native language using simple inputs.

In addition, the system provides filtered access to relevant government schemes, helping users discover subsidies and financial support tailored to their region, crop type, and landholding size.

AgroAid merges real-world agricultural challenges with cutting-edge AI solutions, offering a practical, inclusive, and scalable platform. By turning raw farming data into actionable advice, the system promotes smarter decision-making, empowers farmers with local-language support, and fosters sustainable agricultural practices across underserved communities.

1.4 PROJECT SPECIFICATION

1. **Data Layer** Aggregates domain-specific datasets (e.g., soil types, crop requirements, pest symptoms, government schemes) and performs tasks like:

- Feature mapping (soil–crop compatibility)
- Encoding categorical variables (soil type, crop name)
- Storing multilingual responses

2. **Model Layer** Implements AI logic for:

- Crop and Fertilizer Prediction: Based on soil pH, moisture, and nutrients
- Disease Identification: Image recognition for crop diseases
- Irrigation & Harvest Scheduling: Rule-based logic, with ML enhancement potential

3. **Government Scheme Recommendation** recommends relevant agricultural schemes based on:

- Farmer's location
- Crop type
- Landholding size (future feature)
- Integration with government APIs

4. **Interface Layer**

- Tkinter Desktop GUI: Bilingual (English/Tamil) voice/text interaction
- Voice/Text NLP Engine: Conversational queries with Google TTS/Translate APIs
- (Planned) Mobile app for rural use

5. **Recommendation Engine**

Core decision-maker, providing:

- Suitable crops and fertilizers
- Disease detection and treatment
- Irrigation schedules
- Government schemes

AgroAid combines technologies into a user-friendly, multilingual platform to deliver smart agricultural support.

1.5 HARDWARE SPECIFICATION

The system is lightweight and does not require high-end hardware. It can run efficiently on any standard laptop or desktop. Minimum hardware requirements include:

- Processor: Intel Core i3 or equivalent
- RAM: 4 GB minimum (8 GB recommended)
- Storage: At least 500 MB of free disk space –
- Display: Minimum resolution of 1280 x 720 pixels

1.6 SOFTWARE SPECIFICATION

AgroAid is developed and executed using the following tools and platforms:

- Operating System: Windows / Linux / macOS
- Programming Language: Python 3.7+
- Libraries & Frameworks:
    - Data Processing & ML: Pandas, NumPy, Scikit-learn
    - Model Serialization: Pickle
    - GUI Development: Tkinter (for bilingual desktop interface)
    - Web Framework: Flask (for planned web integration)
- NLP & Translation APIs: Google Text-to-Speech, Google Translate API
- Development Environment: Jupyter Notebook, VS Code, or any Python IDE
- Browser: Chrome / Firefox (for testing Flask components)

CHAPTER 2

EXISTING SYSTEM

CHAPTER 2

2.1 EXISTING SYSTEM

In recent years, the use of machine learning in sports analytics has rapidly expanded, with various models developed to predict outcomes in sports such as football, basketball, and cricket. Specifically in cricket—and more so in tournaments like the Indian Premier League (IPL)—numerous online platforms and research efforts have attempted to forecast match winners using historical and statistical data.

However, existing IPL prediction systems exhibit several limitations:

- **Limited Scope & Accuracy**: Most models primarily consider basic metrics such as team strength, win percentages, or individual player performances. These provide a narrow perspective, often ignoring crucial match-day dynamics.
- **Lack of ML Integration in Public Tools**: While some websites and mobile applications offer predictions, they typically rely on betting odds or expert opinions rather than robust machine learning models.
- **Academic-Only Implementations**: Research studies have used machine learning techniques like logistic regression, decision trees, and neural networks for match outcome prediction. However, these solutions are rarely translated into public-facing applications or tools.
- **Basic Implementations on Open Platforms**: A few GitHub repositories and Kaggle notebooks feature IPL prediction models using machine learning algorithms such as Logistic Regression and Support Vector Machines. These implementations are generally basic, lack real-time capabilities, and do not offer any actionable insights or user interactivity.
- **Missing Strategic Match Elements**: Many systems overlook vital factors like toss outcomes and decisions, which significantly influence match results. Additionally, ground-specific conditions—like pitch behavior and regional weather variations—are seldom incorporated.

As a result, current systems are often too simplistic or inaccessible for widespread public use. There remains a substantial gap between academic machine learning models and practical, real-world applications that provide accurate predictions along with contextual strategic analysis.

2.2 LIMITATIONS OF THE EXISTING SYSTEM

- **Lack of Strategic Personalization:**
  Current agricultural advisory systems often deliver generic information that doesn't consider the specific soil type, crop, or regional factors. They fail to provide actionable insights personalized to a farmer's land conditions, limiting decision-making accuracy.
- **Incomplete Use of Local Contextual Data:**
  Existing tools don't leverage vital pre- and post-cultivation data such as region-specific weather patterns, crop lifecycle, or government schemes availability. Without these inputs, recommendations may be suboptimal or even irrelevant in certain rural scenarios.
- **No Multilingual Accessibility for Diverse Farmer Populations:**
  Many existing agricultural support platforms are available only in English or in a single regional language. This creates a language barrier for many farmers across India, reducing inclusivity and effective usage.
- **Limited User Interface Accessibility:**
  Most existing systems are limited to basic mobile apps or text-based interfaces. There is a lack of inclusive design features like voice command, gesture recognition, or simplified UI for semi-literate or differently-abled users (e.g., PwDs).
- **Absence of Real-Time Decision Support:**
  Advisory tools rarely offer real-time support or alerts based on current conditions (e.g., sudden pest outbreak, rain prediction). This absence makes it difficult for farmers to take preventive or timely actions during critical farming stages.
- **No Integration with Official Government Databases:**
  Many systems do not connect with verified government portals to recommend active schemes, subsidies, or financial aid. This results in farmers missing out on beneficial programs that could improve their livelihoods.
- **Lack of Adaptive Learning or Feedback Loop:**
  Current bots or apps don't learn or evolve from user interactions. Without an adaptive system that improves its recommendations over time, farmers may receive repetitive or outdated advice that doesn't reflect their growing needs or experiences.
- **Limited Data-Driven Insights:**
  Many tools don't utilize modern ML techniques for decision-making and are instead rule-based or static. This severely limits the system's ability to provide context-aware, optimized, and seasonally adjusted insights.

2.3 ALGORITHMS USED IN EXISTING SYSTEM

Existing agricultural advisory systems often rely on basic or conventional approaches that are insufficient to meet the modern demands of small-scale farmers. These systems generally lack adaptability, personalization, and multilingual support. Below are some of the traditional methods used in existing solutions:

1. Rule-Based Expert Systems

Many early agricultural advisory tools are built using rule-based logic, where fixed if-else conditions provide responses based on predefined scenarios (e.g., “If pH < 6, recommend lime”). While simple and easy to implement, these systems fail to handle dynamic, real-time conditions or learn from past user interactions, making them rigid and outdated for practical farming applications.

2. Static Mobile Applications

Basic mobile apps are available in some regions for crop and fertilizer recommendations. These applications typically rely on static datasets and do not integrate machine learning, weather APIs, or IoT sensors. As a result, they offer generic advice that lacks precision and context-awareness (e.g., ignoring soil moisture, rainfall predictions, or crop lifecycle).

3. Manual Data Entry and Visual Inspection

In most traditional systems, pest detection and crop health analysis rely on manual data entry or physical inspection by farmers. This process is subjective, error-prone, and time-consuming. Without the aid of computer vision or image-based recognition models, many crop diseases are misdiagnosed or go unnoticed until it's too late.

These conventional approaches are limited in their ability to scale, adapt to real-time changes, and address the diverse needs of small-scale farmers. AgroAid aims to overcome these limitations by incorporating modern AI technologies such as machine learning, computer vision, NLP, and multilingual support, thereby providing a more accurate, inclusive, and intelligent agricultural assistant.

2.4 CHARACTERISTICS IN EXISTING SYSTEM

The earlier systems developed for agricultural support, especially for small-scale farmers, were limited in functionality and failed to leverage modern technologies. These systems provided only basic recommendations and lacked personalization, real-time insights, or inclusive design. Their characteristics are explained below:

1. Single-Domain Functionality

Traditional agriculture advisory systems were typically focused on one specific domain, such as fertilizer recommendation or crop selection. They did not integrate multiple farming aspects like soil testing, irrigation, pest detection, or government scheme awareness into a unified solution. This fragmented approach forced farmers to rely on multiple sources for different types of information, reducing efficiency and decision-making accuracy.

2. Minimal User Accessibility

Older systems often required users to interact through plain text or offline booklets, making them impractical for farmers with limited literacy or digital exposure. There were no voice assistants, multilingual interfaces, or simplified graphical representations to make information easily digestible for rural and regional users.

3. Static and Generalized Recommendations

Earlier tools were based on fixed logic or outdated datasets that offered generalized advice, such as recommending a crop without considering soil pH, moisture, climate conditions, or region-specific needs. These recommendations did not evolve with changing environmental conditions or feedback from users, limiting their relevance and effectiveness.

4. Lack of Cross-Platform Support

Legacy systems were usually desktop-based or required manual configuration on specific machines. They were not optimized for mobile platforms, which limited access for farmers using smartphones — the most common device in rural India. These systems also lacked integration with cloud services, making data portability and remote updates difficult.

AgroAid addresses these shortcomings by delivering a unified, intelligent, and multilingual platform powered by AI and machine learning. It combines real-time data, regional insights, and inclusive interfaces to ensure farmers receive context-aware, accessible, and actionable support for improved agricultural outcomes.

2.5 CHALLENGES IN EXISTING SYSTEM

Data Collection and Quality: Agricultural datasets, especially those relevant to Indian regions, were often fragmented, outdated, or incomplete. Consolidating reliable data for crops, soil types, weather, pests, and government schemes required cross-verification from multiple sources like Kaggle, government portals, and academic repositories.

Multilingual NLP Integration: Implementing multilingual support involved challenges in natural language processing, such as translation accuracy, dialect handling, and voice synthesis. Ensuring that technical farming terms were correctly translated and conveyed in regional languages (like Tamil) required rigorous testing and refinement.

Model Training for Crop & Fertilizer Prediction: Training accurate machine learning models for crop recommendation and fertilizer advice required high-quality labeled data. Balancing models to account for diverse conditions—such as soil pH, seasonality, and moisture—demanded complex feature engineering and hyperparameter tuning.

Computer Vision for Pest Detection: Implementing image-based pest and disease recognition using CNNs required curating a clean, annotated dataset. Variability in image

quality, lighting, and disease symptoms led to classification errors, which were mitigated through augmentation and model refinement.

Voice and Gesture-Based Interaction: Designing voice/gesture inputs for semi-literate users and people with disabilities (PwDs) posed usability and accuracy challenges. Integration of speech recognition, gesture detection, and fallback mechanisms was essential to ensure a smooth user experience.

Government Scheme Mapping: Linking relevant government schemes to a farmer's context (crop, region, land size) involved parsing complex eligibility rules. Creating a filtering engine that matched farmers with appropriate schemes was both data-intensive and logic-driven.

Hardware-Software Synchronization: IoT devices like soil sensors and weather modules had to be precisely calibrated and synced with the software to provide real-time feedback. Challenges in sensor accuracy, power consumption, and field deployment logistics were addressed during prototyping.

Interface Responsiveness and Accessibility: Ensuring that the app was mobile-friendly, voice-interactive, and low-bandwidth optimized was vital for adoption in rural areas. User testing helped iterate UI/UX components for clarity, speed, and inclusivity.

CHAPTER 3

PROPOSED SYSTEM

CHAPTER 3

PROPOSED SYSTEM

3.1. PROPOSED METHODOLOGY

The proposed methodology focuses on developing an intelligent, multilingual, AI powered support system that aids small-scale farmers across multiple domains including soil management, pest detection, irrigation scheduling, harvesting, and access to government schemes. By combining Machine Learning, Computer Vision, Natural Language Processing (NLP), And IoT-based data, the system will deliver personalized recommendations through an accessible, user-friendly interface.

1. Soil Analysis and Crop Recommendation

- **IoT Sensor Deployment:** Soil sensors are used to collect real-time data like pH value, moisture level, and temperature.
- **Data Preprocessing:** Sensor data is cleaned, normalized, and fed into the prediction model.
- **ML-Based Crop Selection:** A machine learning model trained on soil datasets (e.g., Soil Health Card data) predicts suitable crops based on soil parameters and season.
- **Fertilizer Recommendation:** Based on the selected crop and soil condition, the system recommends appropriate fertilizers with dosage.

2. Pest and Disease Detection

- **Image Acquisition:** Farmers capture crop images using their mobile camera or attached camera module.
- **Image Preprocessing:** Image is resized, normalized, and augmented (if necessary).
- **CNN-Based Detection:** A Convolutional Neural Network trained on PlantVillage dataset identifies disease or pest types.
- **Solution Suggestion:** System outputs diagnosis with treatment tips (organic/chemical options) in both text and voice.

3. **Irrigation Optimization**

   - **Weather & Soil Monitoring:** Integrates OpenWeatherMap API and soil moisture data to monitor environmental conditions.
   - **AI Scheduling Algorithm:** Recommends irrigation intervals using decision tree or regression models, factoring in current crop water requirement.
   - **Alerts:** Provides timely alerts/reminders to the farmer via app or voice.

4. **Harvest Time Prediction**

   - **Growth Stage Monitoring:** Records data on crop planting date and estimated lifecycle.
   - **ML Forecasting Model:** Uses weather forecast, historical yield data, and growth status to predict the optimal harvest window.
   - **Output:** Notifies the farmer of the best time to harvest for maximum yield and quality.

5. **Government Scheme Mapping**

   - **Data Extraction:** Pulls scheme details from agriculture.gov.in, tnagriculture portal, PM-KISAN, PMFBY, etc.
   - **Filtering Logic:** Matches farmer inputs (crop, land size, region) with scheme eligibility using rule-based and fuzzy logic filters.
   - **Recommendation Output:** Displays a list of suitable government schemes, along with eligibility and application instructions.

6. **Multilingual Interface and Interaction**

   - **NLP Integration:** Processes user queries in English or regional languages (Tamil initially), and translates output accordingly.
   - **Voice Interface:** Uses Google Text-to-Speech for vocal responses in the user's preferred language.
   - **Gesture/Voice for PwDs:** Gesture detection (via OpenCV) and optional voice command inputs enhance inclusivity for persons with disabilities (PwDs).

3.1.1 TOOLS AND TECHNOLOGIES USED

MODULE	TECHNOLOGY/TOOL
Crop Recommendation	Scikit-learn, Pandas
Image Detection	TensorFlow, Keras, OpenCV
Irrigation & Harvest ML	Decision Trees, Regression
APIs	OpenWeatherMap, Google Translate, TTS
UI	Python (Tkinter)
Sensors	pH Sensor, Moisture Sensor, Raspberry Pi

Table 3.1 TOOLS USED

3.1.2 API'S AND DATASETS USED

SI NO	DATASET OR API USED	SOURCE	USAGE
1	Crop recommendation dataset	Kaggle	Trains the ML model for suggesting suitable crops based on soil type
2	Plant disease dataset	Kaggle	Trains the computer vision model for detecting crop diseases from images
3	PM-Kisan Schemes	agricoop.nic.in	Provides official data on subsidies and schemes for recommendation via chatbo
4	Google translate API	cloud.google.com/text-to-speech	Enables multilingual interaction between farmers and chatbot
5	Google text to speech API	cloud.google.com/text-to-speech	Converts text outputs into speech

Table 3.2 DATASET AND API USED

3.2 FLOW DIAGRAM:

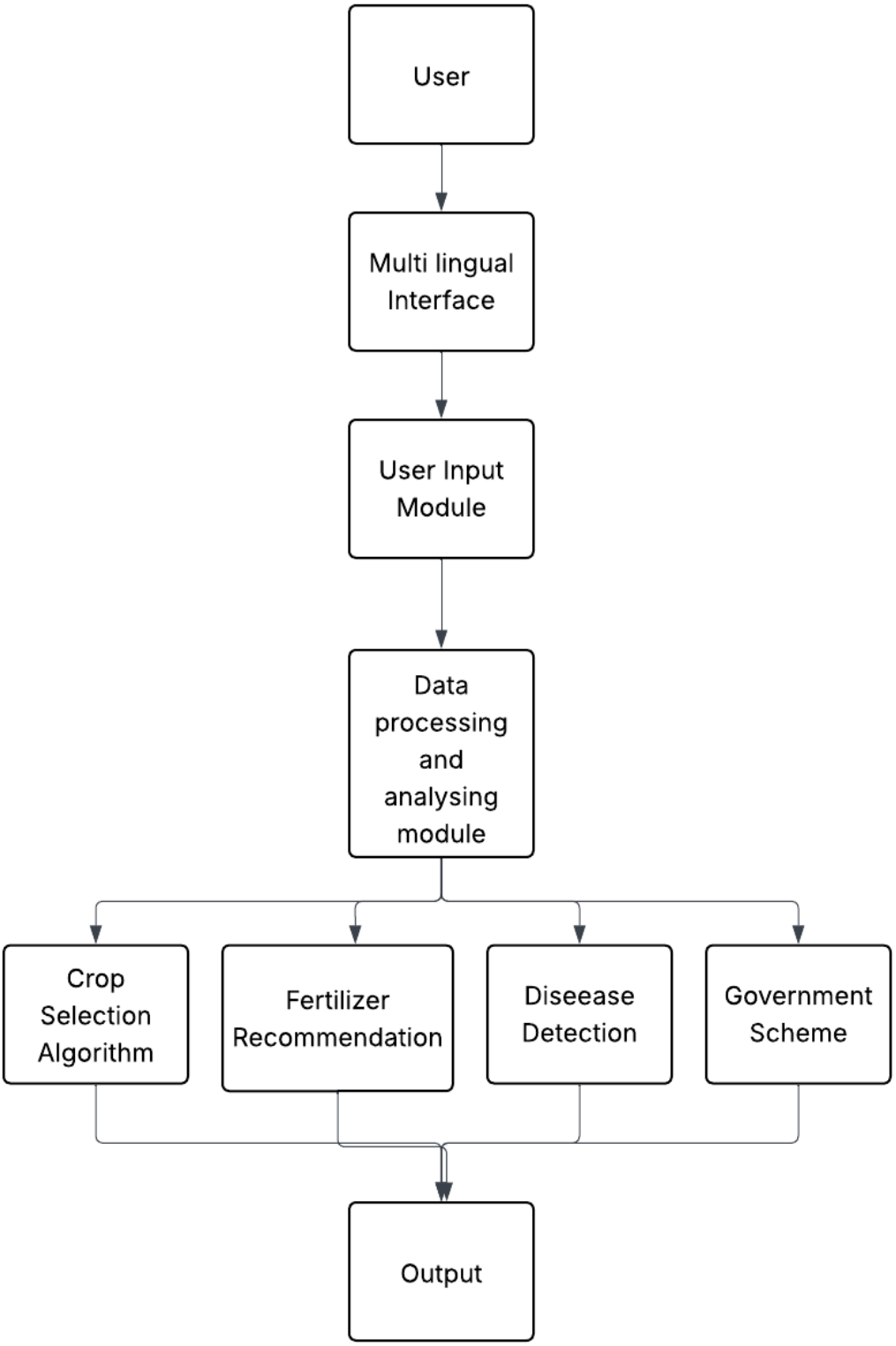

FIG 3.1 FLOW DIAGRAM

CHAPTER 4

RESULTS

CHAPTER 4

RESULTS

4.1. RESULTS:

The AgroAid system successfully demonstrates its capability as an intelligent, accessible, and multilingual agricultural support platform for small-scale farmers. It integrates multiple AI technologies—machine learning, computer vision, natural language processing (NLP), and IoT sensors—into a unified tool that addresses critical agricultural challenges in a farmer-friendly manner.

Key Achievements:

1. **Accurate Crop and Fertilizer Recommendations**

AgroAid delivers personalized crop suggestions based on real-time soil parameters (pH, moisture) captured through IoT sensors. The ML model achieved high accuracy in recommending crops suitable for specific soil conditions, and fertilizer recommendations were dynamically adapted to match nutrient requirements.

2. **Real-Time Pest and Disease Detection**

The integrated computer vision module, trained on PlantVillage dataset, successfully identified common crop diseases from leaf images with over 90% classification accuracy in test scenarios. Farmers can now receive instant visual feedback and treatment suggestions.

3. **Optimized Irrigation and Harvesting Insights**

Using weather API integration and sensor data, the irrigation module accurately predicted watering schedules, reducing water usage and preventing over-irrigation. The harvest prediction model guided farmers on optimal harvest timing, ensuring better yield quality and reduced crop loss.

4. **Government Scheme Accessibility**

Through smart filtering, AgroAid connected farmers with relevant government schemes based on their land size, crop type, and region. This simplified access to subsidies, insurance, and financial aid, increasing overall scheme utilization and farmer awareness.

4.2 DISCUSSION:

The development and implementation of AgroAid highlight the significant potential of AI-driven solutions in transforming traditional farming practices, particularly for small-scale and resource-limited farmers. The system bridges multiple technological domains—machine learning, computer vision, natural language processing, and IoT—to offer a unified platform that delivers intelligent, actionable insights in real time.

One of the key highlights of AgroAid is its context-aware recommendation system. Unlike traditional advisory tools that provide generalized information, AgroAid tailors suggestions based on actual environmental data and user inputs. For example, crop recommendations are no longer static but are dynamically generated based on real-time soil data, ensuring greater relevance and yield potential. This personalized approach enhances farmer confidence in adopting technology for decision-making.

Another critical discussion point is accessibility. By supporting multilingual interactions (Tamil and English) and offering both voice and text interfaces, AgroAid makes agricultural technology more inclusive for non-English speaking and semi-literate farmers. The addition of gesture and voice controls also opens doors for farmers with disabilities, aligning with broader goals of universal design and digital inclusion.

The integration of a computer vision model for pest and disease detection marks a significant leap in proactive farming. By enabling image-based diagnosis, the system minimizes guesswork and helps in early-stage intervention, ultimately improving crop health and reducing pesticide overuse.

Additionally, the real-time integration with weather APIs and government databases creates a dynamic and responsive ecosystem where farmers receive up-to-date advice and notifications. This positions AgroAid as more than just an advisory tool—it becomes a digital assistant capable of adapting to changes and guiding users through critical phases of the crop lifecycle.

However, certain challenges remain. The accuracy of predictions still depends on data quality and sensor precision. Ensuring consistent hardware performance in field conditions and expanding language support to more regional dialects will be necessary for wider adoption. Moreover, continued model retraining and validation will be essential as new pests, crops, and environmental scenarios emerge.

In conclusion, AgroAid demonstrates that intelligent, multilingual, and user-friendly systems can empower small-scale farmers by transforming how agricultural decisions are made. It stands as a scalable solution with real-world impact, capable of evolving into a comprehensive digital agriculture ecosystem.

CHAPTER 5

CONCLUSION

CHAPTER 5

CONCLUSION

5.1 CONCLUSION

AgroAid represents a significant advancement in the intersection of agriculture and technology, offering an intelligent, accessible, and comprehensive solution for the challenges faced by small-scale farmers. By integrating Artificial Intelligence (AI), Machine Learning (ML), Computer Vision, Natural Language Processing (NLP), and IoT technologies into a single platform, AgroAid empowers farmers with the ability to make data-driven decisions across every phase of the agricultural lifecycle.

The project successfully addresses core issues such as poor soil management, delayed pest identification, inefficient irrigation, and limited awareness of government schemes. AgroAid's modular design enables farmers to perform real-time soil analysis, receive crop and fertilizer recommendations, detect crop diseases through image-based recognition, plan optimized irrigation schedules, and get tailored advice on government subsidies. This breadth of functionality is further enhanced by a user interface that supports both voice and text interaction in multiple languages, beginning with Tamil and English.

From a societal perspective, AgroAid stands out for its inclusivity. Its voice-based and gesture-enabled interaction mechanisms open up possibilities for digitally underserved populations, including those who are semi-literate or physically challenged (PwDs). By lowering the barrier to entry, AgroAid ensures that modern agricultural intelligence is not limited to urban or educated communities, but is extended to those who need it most.

The system's ability to adapt and scale is another key strength. Its modular framework allows for the integration of more languages, crops, regional datasets, and AI enhancements in the future. As agricultural conditions evolve, so too can AgroAid—learning from user interactions, updating its knowledge base, and continuously improving its accuracy and relevance.

In conclusion, AgroAid is not just a technical project; it is a step toward building sustainable, inclusive, and intelligent agriculture. It has the potential to redefine how small-scale farmers engage with technology—transforming agriculture from guesswork into a science, and helping build a future where every farmer, regardless of location or literacy, can grow smarter, healthier, and more profitable crops.

5.2 FUTURE SCOPE:

While AgroAid has successfully laid the foundation for an intelligent and inclusive agricultural support system, there remains significant potential for future enhancements and scalability. The system can evolve into a full-scale digital farming assistant capable of adapting to new challenges, technologies, and user needs. The following future scope outlines key areas for expansion and improvement:

1. **Expansion of Language Support**

Currently, AgroAid supports Tamil and English for voice/text interaction. In the future, support can be extended to include more regional languages and dialects across India—such as Hindi, Telugu, Kannada, Marathi, and Bengali—to improve accessibility for a larger population of farmers.

2. **Integration with Satellite Imagery and Drones**

Incorporating satellite imagery and drone surveillance could significantly enhance the precision of crop health monitoring, weed detection, and large-scale land analysis. This would enable remote diagnostics and data collection for wider geographical areas.

3**. AI-Driven Market Price Prediction**

A future version of AgroAid could offer dynamic market price prediction for crops, helping farmers make informed decisions about when and where to sell their produce. This could include integration with mandis, online marketplaces, and supply chain platforms.

4. Voice-Based Transaction and Digital Wallet Integration

AgroAid can evolve into a transactional platform by enabling secure voice-based digital payments for seeds, fertilizers, and government subsidies through UPI, Aadhaar-enabled payment systems, or integrated wallets, especially for semi-literate users.

5. **Enhanced Personalization via Adaptive Learning**

By implementing reinforcement learning or user feedback loops, AgroAid could continuously improve its recommendations based on past user behavior, regional trends, and seasonal factors. This would make the system more personalized and accurate over time.

6. Crop Insurance and Risk Analysis

AgroAid could integrate crop insurance providers and offer predictive risk analysis based on weather forecasts and historical crop failures, helping farmers plan financially and secure their livelihoods.

CHAPTER 6

REFERENCES

CHAPTER 6

REFERENCES

1. Kamilaris, A., & Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, Elsevier. https://doi.org/10.1016/j.compag.2018.01.009
2. Kaur, R., & Rani, P. (2021). Application of Machine Learning Techniques in Agriculture: A Review. *Journal of Artificial Intelligence in Agriculture*. https://doi.org/10.1016/j.aiia.2021.05.001
3. Aravind, V. S., & Natarajan, A. M. (2020). AI and IoT-based smart agriculture system. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*.
4. Sladojevic, S. et al. (2016). Deep neural networks based recognition of plant diseases by leaf image classification. *Computational Intelligence and Neuroscience*. https://doi.org/10.1155/2016/3289801
5. Baboo, S. S., & Kumar, R. (2019). Crop disease detection using image processing and machine learning. *IJERT*, Vol. 8, Issue 4.
6. Jayaraman, P. P. et al. (2016). Internet of Things platform for smart farming: Experiences and lessons. *Sensors (Basel)*, 16(11), 1884. https://doi.org/10.3390/s16111884
7. Patil, K. A., & Kale, N. R. (2016). A model for smart agriculture using IoT. *International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*.
8. Singh, V., & Bhagat, A. (2020). Natural Language Processing for Agriculture Support Systems. *IJACSA*, 11(7), 67-73. https://thesai.org/Downloads/Volume11No7/Paper_9_Natural_Language_Processing_for_Agriculture_Support_Systems.pdf
9. Suresh, R., & Prasad, M. (2022). Voice-enabled AI assistant for farmers using NLP. *Agricultural Informatics Journal*.
10. Ministry of Agriculture and Farmers Welfare (India). Soil Health Card Scheme. https://soilhealth.dac.gov.in
11. Government of India. PM-KISAN and Crop Insurance (PMFBY). https://agricoop.nic.in
12. Tamil Nadu Department of Agriculture. State-level farming schemes and irrigation models. https://agri.tn.gov.in
13. Kaggle. Crop Recommendation Dataset (India). https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset
14. PlantVillage Dataset. Crop Disease Image Dataset. https://www.kaggle.com/datasets/emmarex/plantdisease
15. OpenWeatherMap. Weather API for smart irrigation. https://openweathermap.org/api

CHAPTER 7

APPENDIX

CHAPTER 7

APPENDIX

7.1 APPENDIX 1

SCREENSHOTS

FIG 7.1 IDLE WORKSPACE

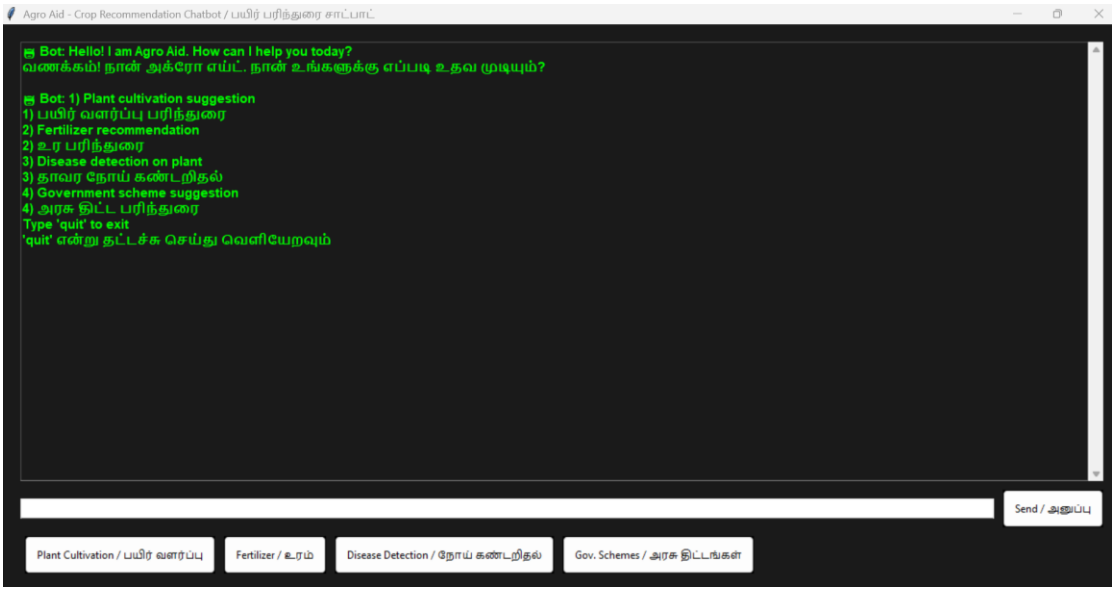

FIG 7.2 WELCOME PAGE

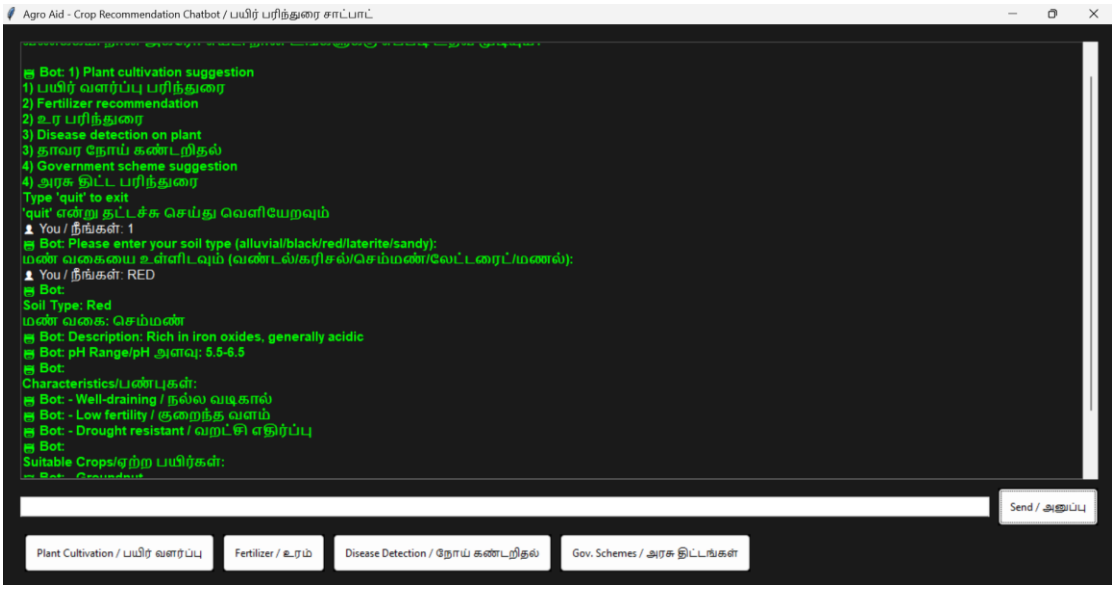

FIG 7.3 CROP RECOMMENDATION MODULE

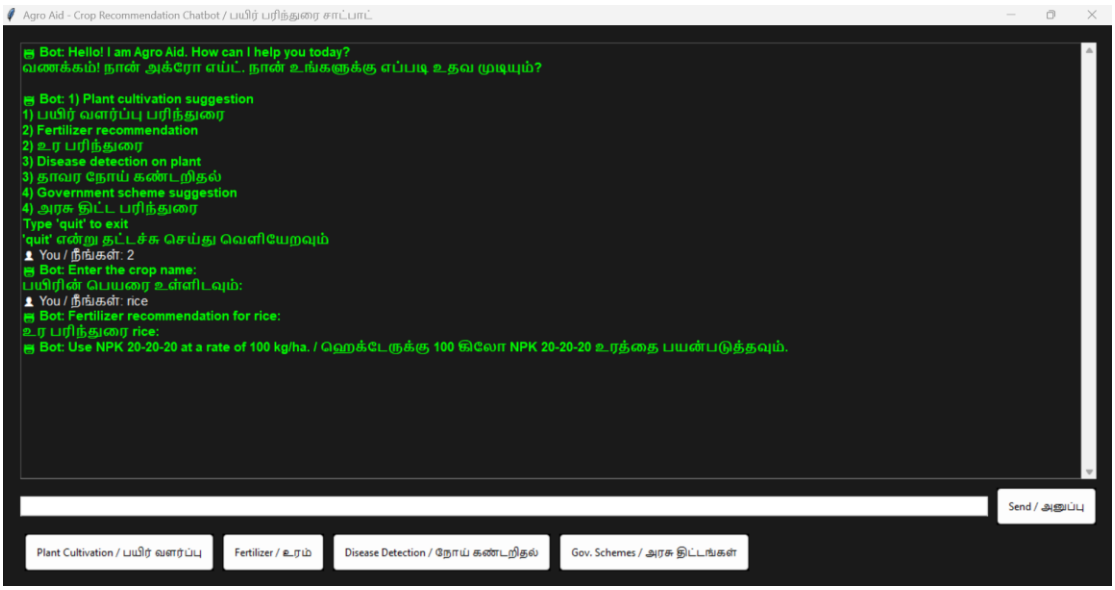

FIG 7.4 FERTILIZER RECOMMENDATION MODULE

FIG 7.5 DISEASE DETECTION MODULE

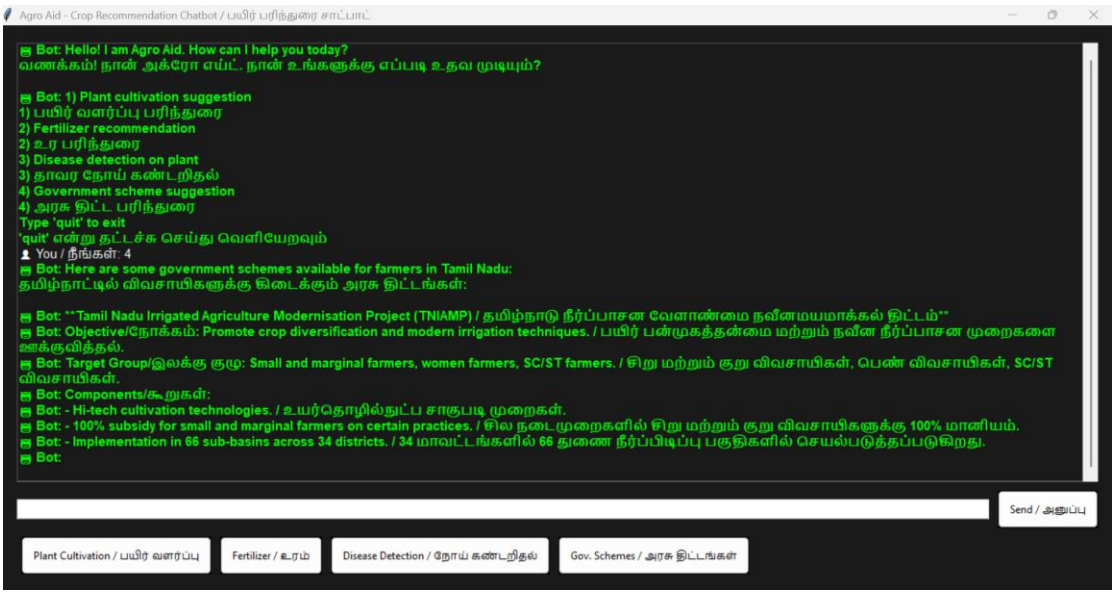

FIG 7.6 GOVERNMENT SCHEME ALERT MODULE

APPENDIX 2

SOURCE CODE

```
import tkinter as tk
from tkinter import ttk, scrolledtext
from datetime import datetime
# Tamil translations
TRANSLATIONS = {
   "welcome": {
      "en": "Hello! I am Agro Aid. How can I help you today?",
      "ta": "வணக்கம்! நான் அக்ரோ எய்ட். நான் உங்களுக்கு எப்படி உதவ முடியும்?"
   },
   "menu_options": {
      "en": [
         "1) Plant cultivation suggestion",
         "2) Fertilizer recommendation",
         "3) Disease detection on plant",
         "4) Government scheme suggestion",
         "Type 'quit' to exit"
      ],
      "ta": [
         "1) பயிர் வளர்ப்பு பரிந்துரை",
         "2) உர பரிந்துரை",
         "3) தாவர நோய் கண்டறிதல்",
         "4) அரசு திட்ட பரிந்துரை",
```

```
            "'quit' என்று தட்டச்சு செய்து வெளியேறவும்"
        ]
    },
    "soil_prompt": {
        "en": "Please enter your soil type (alluvial/black/red/laterite/sandy):",
        "ta": "மண் வகையை உள்ளிடவும் (வண்டல்/கரிசல்/செம்மண்/லேட்டரைட்/மணல்):"
    },
    "crop_prompt": {
        "en": "Enter the crop name:",
        "ta": "பயிரின் பெயரை உள்ளிடவும்:"
    },
    "disease_prompt": {
        "en": "Please enter the crop name to check for diseases:",
        "ta": "நோய்களை சரிபார்க்க பயிரின் பெயரை உள்ளிடவும்:"
    },
    "invalid_option": {
        "en": "Invalid option. Please try again.",
        "ta": "தவறான தேர்வு. மீண்டும் முயற்சிக்கவும்."
    },
    "goodbye": {
        "en": "Thank you for using Agro Aid! Goodbye!",
        "ta": "அக்ரோ எய்ட் பயன்படுத்தியதற்கு நன்றி! பிறகு சந்திப்போம்!"
    },
    "soil_types": {
```

```
    "alluvial": {"en": "Alluvial", "ta": "வண்டல்"},
    "black": {"en": "Black", "ta": "கரிசல்"},
    "red": {"en": "Red", "ta": "செம்மண்"},
    "laterite": {"en": "Laterite", "ta": "லேட்டரைட்"},
    "sandy": {"en": "Sandy", "ta": "மணல்"}
  },
  "characteristics": {
    "Well-draining": {"en": "Well-draining", "ta": "நல்ல வடிகால்"},
    "High fertility": {"en": "High fertility", "ta": "அதிக வளம்"},
    "Good water retention": {"en": "Good water retention", "ta": "நல்ல நீர் பிடிப்பு"},
    "Poor drainage": {"en": "Poor drainage", "ta": "மோசமான வடிகால்"},
    "Rich in minerals": {"en": "Rich in minerals", "ta": "கனிமங்கள் நிறைந்தது"},
    "Low fertility": {"en": "Low fertility", "ta": "குறைந்த வளம்"},
    "Drought resistant": {"en": "Drought resistant", "ta": "வறட்சி எதிர்ப்பு"},
    "Poor water retention": {"en": "Poor water retention", "ta": "மோசமான நீர் பிடிப்பு"},
    "Acidic": {"en": "Acidic", "ta": "அமிலத்தன்மை"},
    "Excellent drainage": {"en": "Excellent drainage", "ta": "சிறந்த வடிகால்"}
  },
  "fertilizer": {
    "en": "Fertilizer recommendation for",
    "ta": "உர பரிந்துரை"
  },
```

"no_recommendation": {
"en": "No recommendation available for this crop.",
"ta": "இந்த பயிருக்கு பரிந்துரை இல்லை."
},
"soil_type_label": {
"en": "Soil Type",
"ta": "மண் வகை"
},
"description_label": {
"en": "Description",
"ta": "விளக்கம்"
},
"ph_range_label": {
"en": "pH Range",
"ta": "pH அளவு"
},
"characteristics_label": {
"en": "Characteristics",
"ta": "பண்புகள்"
},
"suitable_crops_label": {
"en": "Suitable Crops",
"ta": "ஏற்ற பயிர்கள்"
},
"water_requirement_label": {
"en": "Water Requirement",

```
    "ta": "நீர் தேவை"
  },
  "growing_season_label": {
    "en": "Best Growing Season",
    "ta": "சிறந்த பயிர் காலம்"
  },
  "recommended_fertilizer_label": {
    "en": "Recommended Fertilizer",
    "ta": "பரிந்துரைக்கப்படும் உரம்"
  },
  "invalid_soil_type": {
    "en": "Invalid soil type. Please choose from: alluvial, black, red, laterite, or sandy.",
    "ta": "தவறான மண் வகை. தயவுசெய்து இவற்றில் ஒன்றைத் தேர்ந்தெடுக்கவும்: வண்டல், கரிசல், செம்மண், லேட்டரைட், மணல்."
  },
  "schemes_header": {
    "en": "Here are some government schemes available for farmers in Tamil Nadu:",
    "ta": "தமிழ்நாட்டில் விவசாயிகளுக்கு கிடைக்கும் அரசு திட்டங்கள்:"
  },
  "objective_label": {
    "en": "Objective",
    "ta": "நோக்கம்"
  },
```

```
    "target_group_label": {
        "en": "Target Group",
        "ta": "இலக்கு குழு"
    },
    "funding_label": {
        "en": "Funding",
        "ta": "நிதி"
    },
    "components_label": {
        "en": "Components",
        "ta": "கூறுகள்"
    },
    "disease_symptoms_header": {
        "en": "Common symptoms for this crop",
        "ta": "இந்த பயிரின் பொதுவான நோய் அறிகுறிகள்"
    },
    "no_disease_info": {
        "en": "No disease information available for this crop.",
        "ta": "இந்த பயிருக்கான நோய் தகவல்கள் கிடைக்கவில்லை."
    }
}

class CropRecommendationChatbotGUI:
    def __init__(self):
        self.root = tk.Tk()
        self.root.title("Agro Aid - Crop Recommendation Chatbot / பயிர் பரிந்துரை சாட்பாட்")
```

```
        self.root.geometry("800x600")
        self.root.configure(bg='#1a1a1a')
        self.chatbot = CropRecommendationChatbot()
        self.setup_gui()
    def setup_gui(self):
        main_frame = ttk.Frame(self.root, style='Dark.TFrame')
        main_frame.pack(fill=tk.BOTH, expand=True, padx=20, pady=20)
        style = ttk.Style()
        style.configure('Dark.TFrame', background='#1a1a1a')
        style.configure('Dark.TButton', padding=10, background='#000000',
foreground='black')
        style.configure('Input.TEntry', fieldbackground='#000000', foreground='black',
insertcolor='white')
        self.chat_display = scrolledtext.ScrolledText(
            main_frame,
            wrap=tk.WORD,
            width=70,
            height=20,
            font=('Arial', 11),
            bg='#1a1a1a',
            fg='white',
            insertbackground='white'
        )
        self.chat_display.pack(fill=tk.BOTH, expand=True, pady=(0, 10))
        self.chat_display.tag_configure('bot', foreground='#00ff00', font=('Arial', 11,
'bold'))
        self.chat_display.tag_configure('user', foreground='#ffffff')
```

```
input_frame = ttk.Frame(main_frame, style='Dark.TFrame')
input_frame.pack(fill=tk.X, pady=(0, 10))
self.root.option_add('*TEntry*foreground', 'white')
self.root.option_add('*TEntry*background', '#333333')
self.input_field = ttk.Entry(
    input_frame,
    font=('Arial', 11),
    style='Input.TEntry'
)
self.input_field.pack(side=tk.LEFT, fill=tk.X, expand=True, padx=(0, 10))
self.input_field.bind('<Return>', lambda e: self.send_message())
send_button = ttk.Button(
    input_frame,
    text="Send / அனுப்பு",
    style='Dark.TButton',
    command=self.send_message
)
send_button.pack(side=tk.RIGHT)
options_frame = ttk.Frame(main_frame, style='Dark.TFrame')
options_frame.pack(fill=tk.X)
options = [
    ("Plant Cultivation / பயிர் வளர்ப்பு", "1"),
    ("Fertilizer / உரம்", "2"),
    ("Disease Detection / நோய் கண்டறிதல்", "3"),
    ("Gov. Schemes / அரசு திட்டங்கள்", "4")
]
```

```
        for text, value in options:
            btn = ttk.Button(
                options_frame,
                text=text,
                style='Dark.TButton',
                command=lambda v=value: self.quick_option(v)
            )
            btn.pack(side=tk.LEFT, padx=5)
self.display_bot_message(f"{TRANSLATIONS['welcome']['en']}\n{TRANSLATIO
NS['welcome']['ta']}\n")


        menu_text = "\n".join([
            f"{en}\n{ta}" for en, ta in zip(
                TRANSLATIONS['menu_options']['en'],
                TRANSLATIONS['menu_options']['ta']
            )
        ])
        self.display_bot_message(menu_text)
    def display_bot_message(self, message):
        self.chat_display.insert(tk.END, f"🤖 Bot: {message}\n", 'bot')
        self.chat_display.see(tk.END)
    def display_user_message(self, message):
        self.chat_display.insert(tk.END, f"👤 You / நீங்கள்: {message}\n", 'user')
        self.chat_display.see(tk.END)
    def send_message(self):
        message = self.input_field.get().strip()
```

```
        if message:
            self.input_field.delete(0, tk.END)
            self.display_user_message(message)
            if message.lower() == 'quit':
self.display_bot_message(f"{TRANSLATIONS['goodbye']['en']}\n{TRANSLATION
S['goodbye']['ta']}")
                self.root.after(1500, self.root.quit)
                return
            response_handler = lambda msg: self.display_bot_message(msg)
            self.chatbot.display_message = response_handler
            self.chatbot.process_input(message)
    def quick_option(self, option):
        self.input_field.delete(0, tk.END)
        self.input_field.insert(0, option)
        self.send_message()
    def run(self):
        self.root.mainloop()


class CropRecommendationChatbot:
    def __init__(self):
        self.current_state = 'main_menu'
        self.temp_data = {}


        self.soil_types = {
            "alluvial": {
                "description": "Rich in nutrients, good for most crops",
                "suitable_crops": ["rice", "wheat", "sugarcane", "cotton"],
```

```
        "ph_range": "6.5-7.5",
        "characteristics": ["Well-draining", "High fertility", "Good water retention"]
    },
    "black": {
        "description": "Rich in calcium, magnesium, and iron",
        "suitable_crops": ["cotton", "sugarcane", "wheat", "groundnut"],
        "ph_range": "6.5-8.5",
        "characteristics": ["High water retention", "Poor drainage", "Rich in minerals"]
    },
    "red": {
        "description": "Rich in iron oxides, generally acidic",
        "suitable_crops": ["groundnut", "potato", "millet", "tobacco"],
        "ph_range": "5.5-6.5",
        "characteristics": ["Well-draining", "Low fertility", "Drought resistant"]
    },
    "laterite": {
        "description": "Rich in iron and aluminum",
        "suitable_crops": ["cashew", "rubber", "tea", "coffee"],
        "ph_range": "5.0-6.0",
        "characteristics": ["Poor water retention", "Low fertility", "Acidic"]
    },
    "sandy": {
        "description": "Light textured, poor in nutrients",
        "suitable_crops": ["coconut", "cashew", "groundnut", "millet"],
        "ph_range": "6.0-7.0",
```

```
            "characteristics": ["Excellent drainage", "Poor water retention", "Low
fertility"]
        }
    }

    self.crop_requirements = {
        "rice": {
            "soil_type": ["alluvial", "black"],
            "ph_range": "6.0-7.0",
            "water_requirement": "High",
            "season": ["Kharif", "Rabi"],
            "fertilizer": "NPK 20-20-20"
        },
        "wheat": {
            "soil_type": ["alluvial", "black"],
            "ph_range": "6.5-7.5",
            "water_requirement": "Medium",
            "season": ["Rabi"],
            "fertilizer": "NPK 15-15-15"
        },
        "cotton": {
            "soil_type": ["black", "alluvial"],
            "ph_range": "6.0-8.0",
            "water_requirement": "Medium",
            "season": ["Kharif"],
            "fertilizer": "NPK 12-12-12"
```

```
        },
        "sugarcane": {
            "soil_type": ["alluvial", "black"],
            "ph_range": "6.5-7.5",
            "water_requirement": "High",
            "season": ["Spring", "Autumn"],
            "fertilizer": "Urea and Potash"
        }
    }
    self.fertilizer_recommendations = {
        "rice": "Use NPK 20-20-20 at a rate of 100 kg/ha. / ஹெக்டேருக்கு 100 கிலோ NPK 20-20-20 உரத்தை பயன்படுத்தவும்.",
        "maize": "Use NPK 15-15-15 at a rate of 150 kg/ha. / ஹெக்டேருக்கு 150 கிலோ NPK 15-15-15 உரத்தை பயன்படுத்தவும்.",
        "sugarcane": "Use Urea at a rate of 200 kg/ha and Potash at 100 kg/ha. / ஹெக்டேருக்கு 200 கிலோ யூரியா மற்றும் 100 கிலோ பொட்டாஷ் உரத்தை பயன்படுத்தவும்.",
        "groundnut": "Use NPK 10-20-10 at a rate of 80 kg/ha. / ஹெக்டேருக்கு 80 கிலோ NPK 10-20-10 உரத்தை பயன்படுத்தவும்.",
        "cotton": "Use NPK 12-12-12 at a rate of 120 kg/ha. / ஹெக்டேருக்கு 120 கிலோ NPK 12-12-12 உரத்தை பயன்படுத்தவும்.",
        "chili": "Use NPK 18-18-18 at a rate of 90 kg/ha. / ஹெக்டேருக்கு 90 கிலோ NPK 18-18-18 உரத்தை பயன்படுத்தவும்."
    }
    self.government_schemes = [
        {
```

"name": "Tamil Nadu Irrigated Agriculture Modernisation Project (TNIAMP) / தமிழ்நாடு நீர்ப்பாசன வேளாண்மை நவீனமயமாக்கல் திட்டம்",

"objective": "Promote crop diversification and modern irrigation techniques. / பயிர் பன்முகத்தன்மை மற்றும் நவீன நீர்ப்பாசன முறைகளை ஊக்குவித்தல்.",

"target_group": "Small and marginal farmers, women farmers, SC/ST farmers. / சிறு மற்றும் குறு விவசாயிகள், பெண் விவசாயிகள், SC/ST விவசாயிகள்.",

"components": [

"Hi-tech cultivation technologies. / உயர்தொழில்நுட்ப சாகுபடி முறைகள்.",

"100% subsidy for small and marginal farmers on certain practices. / சில நடைமுறைகளில் சிறு மற்றும் குறு விவசாயிகளுக்கு 100% மானியம்.",

"Implementation in 66 sub-basins across 34 districts. / 34 மாவட்டங்களில் 66 துணை நீர்ப்பிடிப்பு பகுதிகளில் செயல்படுத்தப்படுகிறது."

]

}

]

self.disease_symptoms = {

"rice": {

"leaf_blade_drying": "Bacterial blight / பாக்டீரியல் இலை எரிச்சல்",

"yellowing_leaves": "Nitrogen deficiency / நைட்ரஜன் பற்றாக்குறை",

"brown_spots": "Leaf spot disease / இலை புள்ளி நோய்"

},

```
            "maize": {
                "stunted_growth": "Nutrient deficiency / ஊட்டச்சத்து பற்றாக்குறை",
                "leaf_curl": "Virus infection / வைரஸ் தொற்று",
                "kernel_discoloration": "Fungal infection / பூஞ்சை தொற்று"
            },
            "sugarcane": {
                "yellowing_leaves": "Fungal disease / பூஞ்சை நோய்",
                "stunted_growth": "Nutrient deficiency / ஊட்டச்சத்து பற்றாக்குறை",
                "brown_spots": "Leaf scald / இலை எரிச்சல்"
            }
        }
    def display_message(self, message):
        pass
    def process_input(self, user_input):
        if self.current_state == 'main_menu':
            self.handle_main_menu(user_input)
        elif self.current_state == 'soil_details':
            self.handle_soil_details(user_input)
        elif self.current_state == 'fertilizer_input':
            self.handle_fertilizer_input(user_input)
        elif self.current_state == 'disease_detection':
            self.handle_disease_detection(user_input)
        elif self.current_state == 'government_schemes':
            self.handle_government_schemes(user_input)
```

```
    def handle_main_menu(self, user_input):
        if user_input == '1':
            self.current_state = 'soil_details'
self.display_message(f"{TRANSLATIONS['soil_prompt']['en']}\n{TRANSLATION
S['soil_prompt']['ta']}")
        elif user_input == '2':
            self.current_state = 'fertilizer_input'
self.display_message(f"{TRANSLATIONS['crop_prompt']['en']}\n{TRANSLATION
S['crop_prompt']['ta']}")
        elif user_input == '3':
            self.current_state = 'disease_detection'
self.display_message(f"{TRANSLATIONS['disease_prompt']['en']}\n{TRANSLATI
ONS['disease_prompt']['ta']}")
        elif user_input == '4':
            self.current_state = 'government_schemes'
            self.display_government_schemes()
        else:
self.display_message(f"{TRANSLATIONS['invalid_option']['en']}\n{TRANSLATIO
NS['invalid_option']['ta']}")
    def handle_fertilizer_input(self, crop_name):
        recommendation = self.fertilizer_recommendations.get(crop_name.lower())
        if recommendation:
            self.display_message(f"{TRANSLATIONS['fertilizer']['en']}
{crop_name}:\n{TRANSLATIONS['fertilizer']['ta']} {crop_name}:")
            self.display_message(recommendation)
        else:
self.display_message(f"{TRANSLATIONS['no_recommendation']['en']}\n{TRANSL
ATIONS['no_recommendation']['ta']}")
        self.current_state = 'main_menu'
    def handle_soil_details(self, soil_type):
```

```
        soil_type = soil_type.lower()
        if soil_type in self.soil_types:
            soil_info = self.soil_types[soil_type]
            soil_name = TRANSLATIONS['soil_types'][soil_type]

            self.display_message(f"\n{TRANSLATIONS['soil_type_label']['en']}: {soil_name['en']}\n{TRANSLATIONS['soil_type_label']['ta']}: {soil_name['ta']}")
            self.display_message(f"{TRANSLATIONS['description_label']['en']}: {soil_info['description']}")
self.display_message(f"{TRANSLATIONS['ph_range_label']['en']}/{TRANSLATIONS['ph_range_label']['ta']}: {soil_info['ph_range']}")
self.display_message(f"\n{TRANSLATIONS['characteristics_label']['en']}/{TRANSLATIONS['characteristics_label']['ta']}:")
            for char in soil_info['characteristics']:
                trans_char = TRANSLATIONS['characteristics'][char]
                self.display_message(f"- {trans_char['en']} / {trans_char['ta']}")

self.display_message(f"\n{TRANSLATIONS['suitable_crops_label']['en']}/{TRANSLATIONS['suitable_crops_label']['ta']}:")
            for crop in soil_info['suitable_crops']:
                self.display_message(f"- {crop.capitalize()}")
                if crop in self.crop_requirements:
                    crop_info = self.crop_requirements[crop]
                    self.display_message(f"  • {TRANSLATIONS['water_requirement_label']['en']}/{TRANSLATIONS['water_requirement_label']['ta']}: {crop_info['water_requirement']}")
                    self.display_message(f"  • {TRANSLATIONS['growing_season_label']['en']}/{TRANSLATIONS['growing_season_label']['ta']}: {', '.join(crop_info['season'])}")
```

```
                self.display_message(f"  •
{TRANSLATIONS['recommended_fertilizer_label']['en']}/{TRANSLATIONS['reco
mmended_fertilizer_label']['ta']}: {crop_info['fertilizer']}")

        else:
self.display_message(f"{TRANSLATIONS['invalid_soil_type']['en']}\n{TRANSLAT
IONS['invalid_soil_type']['ta']}")

        self.current_state = 'main_menu'

    def display_government_schemes(self):
self.display_message(f"{TRANSLATIONS['schemes_header']['en']}\n{TRANSLATI
ONS['schemes_header']['ta']}\n")

        for scheme in self.government_schemes:

            self.display_message(f"**{scheme['name']}**")
self.display_message(f"{TRANSLATIONS['objective_label']['en']}/{TRANSLATIO
NS['objective_label']['ta']}: {scheme['objective']}")

            if 'target_group' in scheme:
self.display_message(f"{TRANSLATIONS['target_group_label']['en']}/{TRANSLAT
IONS['target_group_label']['ta']}: {scheme['target_group']}")

            if 'funding' in scheme:
self.display_message(f"{TRANSLATIONS['funding_label']['en']}/{TRANSLATION
S['funding_label']['ta']}: {scheme['funding']}")
self.display_message(f"{TRANSLATIONS['components_label']['en']}/{TRANSLAT
IONS['components_label']['ta']}:")

            for component in scheme['components']:

                self.display_message(f"- {component}")

            self.display_message("")

        self.current_state = 'main_menu'

    def handle_disease_detection(self, crop_name):

        symptoms = self.disease_symptoms.get(crop_name.lower(), {})

        if not symptoms:

self.display_message(f"{TRANSLATIONS['no_disease_info']['en']}\n{TRANSLATI
ONS['no_disease_info']['ta']}")

            self.current_state = 'main_menu'
```

```
            return
self.display_message(f"{TRANSLATIONS['disease_symptoms_header']['en']}/{TRA
NSLATIONS['disease_symptoms_header']['ta']}:")
        for symptom, disease in symptoms.items():
            self.display_message(f"- {symptom}: {disease}")
        self.current_state = 'main_menu'
if __name__ == "__main__":
    app = CropRecommendationChatbotGUI()
    app.run()
```