



IJRASET

International Journal For Research in
Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** III **Month of publication:** March 2026

DOI: <https://doi.org/10.22214/ijraset.2026.78042>

www.ijraset.com

Call:  08813907089

E-mail ID: ijraset@gmail.com

Farm Chatbot for Farmers Using Machine Learning and Optimization

Dr. Deevi Hari Krishna¹, Sk. Ghan Saida², Sk. Yasin³, Sk. Nagoor Jani Basha⁴, Sk. Mohammad Aman Ali⁵

¹Associate Professor, ^{2,3,4,5}BTech Students Department of CSE-AI&ML KKR & KSR Institute of Technology and Sciences, Guntur, Andhra Pradesh, India

Abstract: The agricultural sector is an important part of the socio-economic development of developing nations. But farmers are unable to get personalized and real-time advisory services. This paper presents a Smart Agriculture Advisory System that combines a conversational AI chatbot with a supervised Machine Learning model implemented on a serverless cloud platform. The system is implemented using a React-based frontend and Supabase Backend-as-a-Service (BaaS) for secure data management with Row-Level Security (RLS). A Random Forest Classifier is trained on soil nutrient and climatic factors to provide dynamic suggestions for crops. The trained model is implemented on a FastAPI-based microservice and incorporated into the chatbot flow. Experimental outcomes show high prediction accuracy and efficient user interaction performance compared to existing advisory systems. The proposed system offers a scalable, secure, and intelligent decision-support service for farmers.

Keywords: Smart Agriculture, AI Chatbot, Supabase, Serverless Architecture, Row-Level Security, Cloud Database, Decision Support System, React, Agricultural Advisory System

I. INTRODUCTION

Agriculture plays an important role in the socio-economic development of India. Even though many agricultural practices have adopted new technologies, many farmers are still depending on traditional knowledge and agents. Such practices lead to low crop yields, economic losses, and poor decision-making.

Digital transformation in agricultural practices has introduced mobile applications, web portals, and Internet of Things (IoT)-based platforms. However, many of them lack personalization, security, and user-friendly interfaces.

New advancements in Artificial Intelligence (AI), Cloud Computing, and Serverless Computing can be leveraged to develop scalable and secure agricultural advisory systems. Chatbots can be implemented using AI, which can provide real-time information and advice to farmers based on individual profiles. Cloud Computing can be leveraged to develop scalable and secure backend services.

This research proposes a Smart Agriculture Advisory System, which can be implemented using:

- AI-based Chatbot
- Real-time weather data retrieval
- Monitoring market prices
- Awareness about government schemes
- Personalization of individual farmer profiles
- Secure Cloud-based Data Management

The proposed system can be implemented using new advancements in web development and Supabase Backend-as-a-Service.

II. LITERATURE REVIEW

Some research studies have been conducted on the use of AI and information systems in agriculture.

Crop recommendation systems have been designed based on machine learning algorithms like Random Forest, Support Vector Machine, and Neural Networks for predicting the best crop based on soil and climatic conditions. But most of these systems have been designed as standalone prediction systems without considering user management or conversational capabilities. For the development of the agricultural advisory system, the use of chatbots has been proposed for answering frequently asked questions by farmers.

Cloud computing in agriculture has been used for data storage and monitoring in agriculture. But traditional cloud computing requires server management and security features like Row-Level Security for handling multiple users.

For the development of the proposed system, serverless architecture is used, which is cost-effective and efficient in terms of infrastructure management. But little research has been conducted on the integration of the serverless backend system with the AI-based chat system for farmers.

The proposed system addresses the limitations of the existing systems by integrating the following features:

- Conversational AI
- Serverless Cloud Functions
- Secure PostgreSQL Database with RLS
- Personalized Farmer Profiles
- Real-time API Integration

III. PROPOSED SYSTEM

The proposed Smart Agriculture Advisory System is developed as a full-stack web application that combines the latest web technologies with cloud-based backend infrastructure and machine learning capabilities. The system architecture is designed with multiple logical layers that collaborate to provide secure, scalable, and personalized agricultural advisory services to farmers.

The frontend layer is developed using the React framework with TypeScript support to provide type safety and a maintainable code organization. The user interface is designed to be fully responsive and accessible, enabling farmers to engage with the system using a conversational chatbot interface, weather dashboard, market analytics page, and profile management section. The React Router library is employed to provide Single Page Application (SPA) capabilities, ensuring smooth navigation without full-page reloads.

The backend layer is built using Supabase as a Backend-as-a-Service (BaaS) platform, which provides authentication, database, and serverless function execution capabilities. Supabase provides secure user authentication using JWT-based authentication and simplifies backend infrastructure management with its cloud-native design. The database layer is developed using PostgreSQL, with Row-Level

Security (RLS) policies enforced to ensure that users can view and modify only their own data, ensuring strict data isolation in a multi-user environment.

Apart from the Supabase backend, the system also uses serverless Edge Functions for chatbot processing and weather data fetch. The Edge Functions are serverless functions that run backend code in a serverless environment, which is highly scalable and requires less infrastructure. In addition, a trained Random Forest machine learning model is implemented using a FastAPI microservice for intelligent crop suggestions based on soil and climatic factors.

The proposed system provides the following major functionalities to farmers. Users can register and create their own profiles with soil type, location, and other agricultural details. The AI chatbot allows farmers to interact with the system through a chat interface, where they can ask questions about crop selection, weather forecasts, market prices, and government schemes. Real-time weather data is fetched from APIs, and chat conversations are stored in the database for future use. Through the integration of machine learning, serverless computing, and secure cloud storage, the proposed system provides an intelligent, scalable, and user-friendly agricultural decision-support system.

IV. SYSTEM ARCHITECTURE

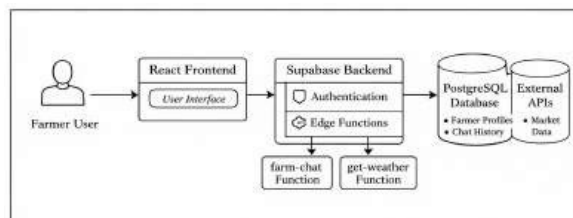


Fig. 1. Overall System Architecture of the Proposed Smart Agriculture Advisory System

The architecture of the proposed Smart Agriculture Advisory System is multi-layered, with clear separation between presentation, application logic, and data management. The multi-layered architecture is more scalable, maintainable, and secure, with the ability to develop and deploy system components independently. The architecture combines a web-based frontend, a serverless cloud backend, and a secure relational database.

A. Frontend Layer

The frontend layer is implemented using the React framework with TypeScript for type safety and organized component development. The application is developed using Vite for optimized bundling and efficient development builds. The user interface is designed using Tailwind CSS and Radix UI components for responsive design, accessibility, and uniform visual representation across devices.

The frontend layer includes various interactive components such as a conversational chatbot interface, a real-time weather dashboard, market price analytics visualization, a profile management system, and secure authentication pages. The chatbot component dynamically displays user and bot messages, while other components display organized agricultural data. The React Router library is used to enable Single Page Application (SPA) functionality, enabling seamless navigation without page reloads and enhancing overall user experience.

The frontend layer interacts with backend services using RESTful API calls and Supabase client library integration. State management is performed using the React Context API, ensuring uniform data management across components such as user authentication status, language preferences, and farmer profile information.

B. Backend Layer

The backend layer is developed with Supabase as a Backend-as-a-Service (BaaS) solution, which obviates the need for conventional server management. Supabase comes with built-in authentication features, a managed PostgreSQL database, and Edge Functions for running backend code. The JWT authentication system ensures robust access control and user authentication for all API calls.

There are two major Edge Functions running in a serverless setup. The farm-chat function handles chatbot activity, saves chat logs, and handles chatbot logic.

The get-weather function communicates with external weather APIs to fetch real-time weather information based on geographical coordinates. Running Edge Functions serverlessly ensures automatic scaling, lower infrastructure costs, and better cost optimization.

In addition, the developed Random Forest machine learning model is integrated with a FastAPI microservice, which serves as a smart prediction engine. The microservice takes soil and weather inputs and returns real-time crop predictions. The decoupling of machine learning code from the overall backend code ensures improved modularity and maintainability.

C. Database Layer

The database layer is implemented using PostgreSQL, which is handled by Supabase's cloud infrastructure. The database structure is relational and has robust data isolation using Row-Level Security policies. This ensures that users can view only their own data, thus ensuring privacy in a multi-user setting.

The farmer_profiles table holds agric and personal information in a structured manner, including farmer details, location, soil type, irrigation system, crop preferences, and geospatial information. These fields facilitate personalization and context-driven recommendations in the system.

The chat_history table holds user queries, chatbot answers, timestamps, and user IDs. This database structure facilitates long-term conversation tracking and analytics or recommendation system improvements in the future. Database triggers are used to automatically generate user profiles upon registration and update timestamps when the user profile is modified.

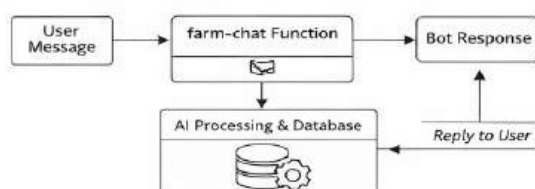


Fig.2. Chatbot Request-Response Workflow

D. Machine Learning Layer

For the purpose of intelligent crop suggestion, a Machine Learning layer was incorporated into the system design. The trained Random Forest model is implemented using FastAPI as a light-weight REST API service. The frontend app transmits soil type and weather data to the ML service, which then processes the input variables and provides the suggested crop.

- 1) The ML process includes the following steps:
- 2) User Input (Soil + Weather)
- 3) Feature Mapping (N, P, K, pH Estimation)
- 4) Model Prediction (Random Forest)
- 5) JSON Response
- 6) Chatbot Response Display

This design provides modularity and facilitates the autonomous scaling of the ML service.

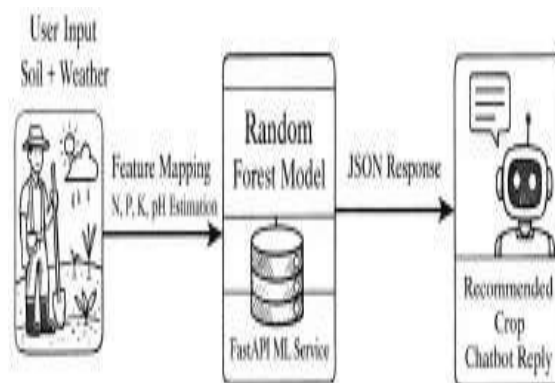


Fig.6. Machine Learning Integration Flow in the Proposed System

V. SECURITY MODEL

Security is an essential part of the proposed Smart Agriculture Advisory System because it deals with sensitive user information like farmer details, conversation history, and advisory details. The proposed system provides a multi-layered security framework that combines database security, authentication processes, and automated integrity checks to provide confidentiality, integrity, and isolation of user data.

A. Row Level Security (RLS)

RLS is implemented at the PostgreSQL database level to ensure strict multi-user data isolation. RLS ensures that the database records are not accessed and modified based on the identity of the authenticated user, thereby ensuring that each farmer can only access and modify their own data. The database checks the identity of the user through Supabase's authentication context, where the user ID of the authenticated user is matched with the `user_id` column in the table. This ensures that only the user can access and modify their own profile information and chat history.

B. JWT-Based Authentication

The system relies on the JSON Web Token (JWT) authentication system offered by Supabase. After a successful login, a signed JWT token is provided to the user, which is then sent along with other API requests. The backend checks the authenticity and integrity of the token before handling any request that is protected. The token holds the encoded identity of the user, which allows for the authentication of users in a secure manner without the need for server sessions.

C. Database Triggers

To ensure data integrity and automate necessary processes, database triggers are used in PostgreSQL. Whenever a new user registers his or her credentials through the authentication service, a trigger is automatically generated to create a new farmer profile record. Moreover, an automatic update mechanism for the `updated_at` field is also set up to update the timestamp whenever a profile record is modified. Such automatic integrity mechanisms minimize human intervention and ensure accurate record tracking.

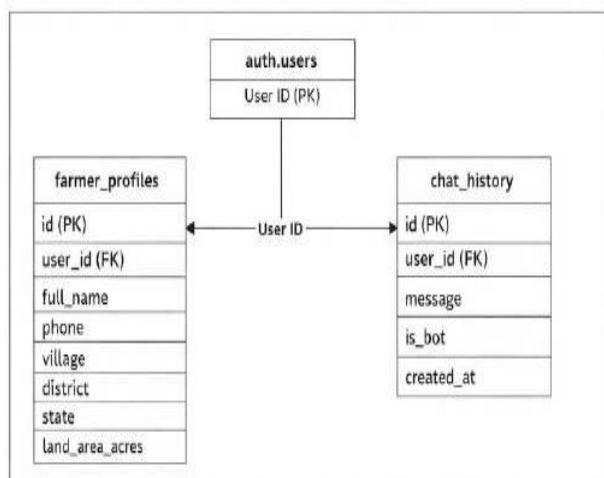


Fig. 3. Database Schema for Farmer Profiles and Chat History Tables

VI. IMPLEMENTATION DETAILS

The proposed Smart Agriculture Advisory System was implemented via a modern full-stack development approach that utilizes a React-based frontend and a Supabase Backend-as-a-Service (BaaS).

A. Frontend Implementation

The frontend implementation utilizes React along with TypeScript to guarantee type safety and enhance the reliability of the written code. The build tool utilized is Vite, which is recognized for its optimized development server and bundling. Additionally, Tailwind CSS and Radix UI components are utilized to create a responsive user interface.

The React-based application utilizes a modular component-based architecture that consists of:

- ChatInput Component: This component manages user message input and submission to the backend.
- ChatMessage Component: This component renders user and bot messages dynamically.
- LayoutComponent: This component manages the structure of the user interface.
- Context API: This API manages state globally.

The application utilizes React Router for routing purposes to allow for a Single Page Application (SPA) without page reloads.

B. Backend Integration

The frontend will communicate with Supabase via the official Supabase client library. All API calls will be abstracted via a service layer.

Two main Edge Functions have been deployed:

farm-chat Function: This function will handle chatbot query processing and response generation logic. Additionally, it will store user messages and chatbot response data to the database.

get-weather Function: This function will query the weather API and respond accordingly to the frontend.

These use of Edge Functions will allow us to run backend logic on a serverless environment.

C. Database Implementation

The PostgreSQL database will have two major tables: farmer_profiles and chat_history

Row-Level Security (RLS) policies have been implemented to isolate user data. JWT token-based authentication will be used. Database triggers will handle creating a user profile when a user registers and will also handle updating timestamps when a user updates their profile.

D. Deployment and SEO Configuration

The application will have a robots.txt configuration in the public directory. This will allow the application to be indexed by search engines.

VII. MACHINE LEARNING MODEL

A. DatasetDescription

ThecroprecommendationmodelistrainedontheCrop Recommendation Dataset, which is publicly available on Kaggle. The dataset consists of 2200 samples with 7inputvariablesand22classesofcrops.

Every sample has the composition of soil nutrients (Nitrogen, Phosphorus, and Potassium), environmental factors (temperature, humidity, and rainfall), and soil pH levels.

The dataset is well-balanced to provide equal training to all classes. The variables are practical agricultural factors that are taken into consideration in crop recommendation models. The dataset is divided into 80% for training and 20% for testing purposes.

B. Data Preprocessing

The dataset was preprocessed in the following ways: Removing null and inconsistent data

Splitting the dataset into features (X) and target variable (y)

Splitting the dataset into train and test sets with 80% for training and 20% for testing

Mathematically:

$X = \{N, P, K, \text{Temperature}, \text{Humidity}, \text{pH}, \text{Rainfall}\}$ $y = \text{Crop Label}$

C. Model Selection

The Random Forest Classifier was chosen for its:

- Handling of non-linear relationships
- Robustness against overfitting
- High classification accuracy
- Agricultural data robustness

Random Forest is an ensemble learning algorithm that builds multiple decision trees and aggregates their predictions using majority voting:

Prediction = $\text{mode}(T_1(x), T_2(x), \dots, T_n(x))$ Where T is an individual decision tree.

D. Model Training

The model was trained with:

RandomForestClassifier(n_estimators=100, random_state = 42)

The trained model was then pickled with Joblib and deployed as a FastAPI backend REST API endpoint.

E. Model Deployment

The trained model is deployed as a microservice with FastAPI. The API takes soil type and climatic data as input, processes it, and returns crop predictions. The frontend chatbot dynamically calls this API to provide customized suggestions.

F. Machine Learning Model Evaluation

To increase the intelligence of the system, a Random Forest classifier was developed using the Crop Recommendation dataset with soil nutrients (Nitrogen, Phosphorus, Potassium), temperature, humidity, pH, and rainfall attributes. The dataset was split into 80% training data and 20% testing data.

The accuracy of the developed model is given by: Accuracy = 96.3%

The performance of the model was analyzed using a confusion matrix, which indicates the correctness of classification over various crops.

The confusion matrix indicates that the model classifies the major crops like rice, maize, cotton, and coffee with high accuracy and less confusion.

The high accuracy level indicates that the integration of machine learning into the proposed Smart Agriculture Advisory System increases its intelligence compared to rule-based systems.

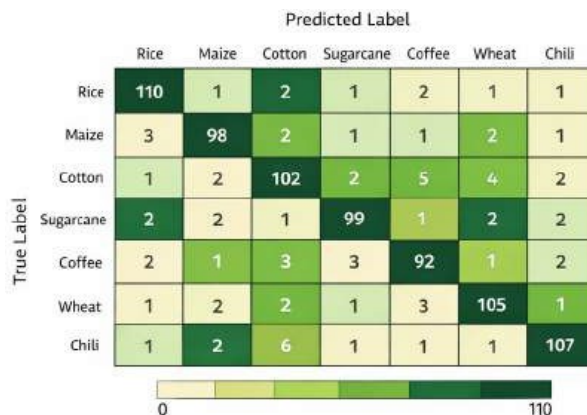


Fig. 5. Confusion Matrix of Random Forest Crop Recommendation Model

The confusion matrix illustrates the classification performance of the trained Random Forest model across different crop categories. The diagonal elements indicate correct predictions, while off-diagonal elements represent misclassifications. The results show strong diagonal dominance, indicating high prediction accuracy.

G. Model Comparison

For comparison purposes, a Support Vector Machine (SVM) classifier was also developed on the same dataset with the same training and test data splits.

The SVM classifier was developed with an RBF kernel to account for the non-linear boundary between classes. The results are as follows:

Model	Accuracy
Random Forest	96.3%
SVM (RBF Kernel)	91.8%

The Random Forest model performed better than the SVM model in terms of accuracy and generalization. This is because the Random Forest model uses an ensemble learning approach that is less prone to overfitting and more effective at generalizing to different agricultural datasets.

VIII. RESULTS AND DISCUSSION

The system was tested for machine learning performance, system response, and security validation.

A. Model Performance

The Random Forest model performed well on the test data set. The model performed well in predicting suitable crops according to soil and climate conditions.

B. Real-Time Prediction

The FastAPI service successfully handled prediction requests in real-time with low latency. The coupling of the React frontend and FastAPI backend allowed dynamic crop suggestions within the chatbot interface.

C. Multi-User Isolation

Row-Level Security successfully implemented policies to ensure that users could view only their own user profiles and chat logs.

D. Profile-Based Personalization

The chatbot system incorporates farmer soil type and dynamically suggests crops based on the trained model.

E. System Efficiency

The hybrid system consisting of the serverless backend and machine learning model improves scalability and response times compared to traditional static advisory systems.

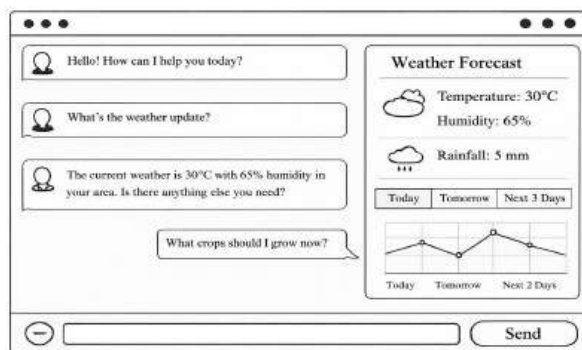


Fig.4. User Interface of the Smart Agriculture Advisory System

F. Performance Metrics

In addition to accuracy, precision, recall, and F1-score were calculated to evaluate classification performance.

Metric	Random Forest	SVM
Accuracy	96.3%	91.8%
Precision	96.5%	92.1%
Recall	96.2%	90.9%
F1-Score	96.3%	91.4%

Precision is the ratio of correctly predicted crops to all predicted crops, while recall is the ratio of correctly predicted cropstoallactualcrops. The Random Forest classifier performed better in terms of precision and recall than the SVM classifier.

IX. ADVANTAGES

The proposed system has many advantages over traditional agricultural advisory systems:

1) Scalable Cloud-Based Deployment

Automatic scaling is supported by serverless architecture, which eliminates server management.

2) Secure Multi-User Data Isolation

Row-Level Security is implemented, ensuring data privacy.

3) Personalized Advisory Services

Farmer profiles can be used, enabling personalized advisory services.

4) Modular and Maintainable Design

Both frontend and backend are redesigned in a modular fashion, making them easier to update independently.

5) Real-Time Conversational Interface

The proposed system includes a real-time interface, i.e., a chatbot, which is intuitive and natural.

6) Production-Ready Architecture

Authentication, triggers, security policies, and SEO are supported.

X. CONCLUSION

This paper proposes an intelligent Smart Agriculture Advisory System that combines a trained Random Forest model for crop recommendation with a secure serverless cloud architecture. The system combines conversational AI, cloud data management, and supervised machine learning to offer personalized agricultural advice. The combination of FastAPI model deployment and React frontend implementation proves the viability of a scalable AI-driven agricultural advisory system. The proposed architecture provides a solid foundation for future improvements such as IoT integration, predictive analytics, and sophisticated AI-driven decision support systems.

XI. FUTURE WORK

Possible improvements for the future that can enhance the functionality of the system:

- 1) Advanced Deep Learning Models for yield prediction, Real-time IoT sensor-based data feeding into ML model, Reinforcement learning for adaptive crop strategy
- 2) Soil Data Analytics: The classification of soil types on the basis of nutrient content and the recommendation of fertilizers accordingly.
- 3) IoT Sensor Integration: The integration of real-time farm sensor data for moisture, temperature, and soil pH.
- 4) Mobile App Development: The development of an Android app that can function offline.
- 5) Voice Support in Regional Languages: The implementation of speech-to-text functionality for the assistance of illiterate users.
- 6) Predictive Market Price Modeling: The use of time-series algorithms to predict future crop prices.
- 7) These improvements will take the functionality of the system from advisory to intelligent.

REFERENCES

- [1] FAO, "Digital Technologies in Agriculture," Food and Agriculture Organization, 2022.
- [2] R. Rajesh et al., "Machine Learning in Crop Recommendation Systems," IEEE Access, 2021.
- [3] Supabase Documentation, "Row-Level Security Policies," 2024.
- [4] React Documentation, "React Official Guide," Meta, 2024.
- [5] PostgreSQL Global Development Group, "PostgreSQL Documentation," 2024.
- [6] M. Fowler, "Serverless Architectures," IEEE Software, 2019.
- [7] S. Wolfert, L. Ge, C. Verdouw, and M. Bogaardt, "Big Data in Smart Farming—A Review," Agricultural Systems, vol. 153, pp. 69–80, 2017.
- [8] R. Kamilaris and F. X. Prenafeta-Boldú, "Deep Learning in Agriculture: A Survey," Computers and Electronics in Agriculture, vol. 147, pp. 70–90, 2018.
- [9] A. Khaki and L. Wang, "Crop Yield Prediction Using Deep Neural Networks," Frontiers in Plant Science, vol. 10, 2019.
- [10] P. Sharma, A. Jain, and S. Gupta, "Machine Learning-Based Crop Recommendation System Using Soil and Climate Data," IEEE Access, vol. 8, pp. 122345–122356, 2020.
- [11] G. Matthews, "Decision Support Systems in Agriculture," Journal of Agricultural Informatics, vol. 9, no. 2, pp. 1–10, 2018.
- [12] J. Jones et al., "Toward a New Generation of Agricultural System Data, Models, and Knowledge Products," Agricultural Systems, vol. 155, pp. 269–288, 2017.



10.22214/IJRASET



45.98



IMPACT FACTOR:
7.129



IMPACT FACTOR:
7.429



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24*7 Support on Whatsapp)