Automation, Control and Intelligent Systems
2025, Vol. 13, No. 2, pp. 22-30
https://doi.org/10.11648/j.acis.20251302.11

Research Article

AI-Driven Agricultural Advisor: Real-Time, Multilingual Support with Lang Graph & Weather APIs

Neha Bansal, Bhawna Singla*

School of Computer Science and Engineering, Geeta University, Panipat, India

Abstract

This research introduces the development of a smart AI agent using LangGraph to provide real-time, location-specific agricultural recommendations to farmers. Leveraging the integration of AgroMonitoring weather APIs, the system collects and analyzes live weather data such as temperature, rainfall, humidity, and soil moisture, which are crucial for agricultural decision-making. By combining this weather data with decision tree-based algorithms, the AI agent predicts optimal crop choices, suggests irrigation practices, and offers rainwater harvesting strategies tailored to the farmer's specific region. The use of decision trees ensures that these recommendations are not only data-driven but also interpretable, allowing farmers to understand the reasoning behind the advice. Furthermore, the AI agent incorporates multilingual support to enhance accessibility. It translates agricultural advice into local languages, such as Hindi, Tamil, and Marathi, making the system inclusive and usable for a diverse range of farmers across different regions of India. This feature significantly reduces language barriers, enabling farmers from various linguistic backgrounds to engage with the technology more effectively. The paper delves into the workflow of the system, from data collection to backend implementation using LangGraph, and discusses its broader impact on promoting sustainable farming practices. By providing farmers with timely, personalized insights, the AI agent empowers them to make informed decisions that improve crop yields, optimize resource use, and contribute to long-term agricultural sustainability. Ultimately, this approach aims to foster a more resilient and productive agricultural landscape, benefiting both farmers and the environment.

Keywords

Agriculture, Crop Sowing, Weather Conditions, Weather API, AI-Driven Decision Tree Algorithm, LangGraph, AI Agent

1. Introduction

Agriculture is intrinsically linked to the environment, with factors such as temperature, humidity, soil pH, and rainfall determining whether a crop will thrive in a specific region. In traditional agricultural practices, farmers often rely on generational knowledge to decide which crops to plant. However, in today's fast-changing world, this approach may no longer suffice due to the increasing unpredictability of weather patterns and the accelerating pace of climate change. With technological advancements, particularly in Artificial Intelligence (AI), farmers now have access to more accurate, data-driven tools that can make decisions with minimal human intervention. AI algorithms, when combined with live weather data, can offer personalized crop recommendations, optimized irrigation strategies, and sustainable farming practices based

*Corresponding author: Bhawna_singla@yahoo.com (Bhawna Singla)

Received: 9 May 2025; **Accepted:** 21 May 2025; **Published:** 19 June 2025

on both current and forecasted weather conditions. This paper introduces an AI-driven crop recommendation system that leverages LangGraph [1], a composable language model framework, alongside weather data APIs such as AgroMonitoring [2, 3]. The system employs decision tree algorithms to predict the most suitable crops for any given location, and its multilingual capabilities ensure that the recommendations are accessible to farmers in their native language. The unique aspect of this framework is that once the initial process is triggered, the entire sequence of operations is automated. From data fetching and processing to crop recommendation and advisory delivery, all tasks are handled by AI agents with no human intervention required after initiation. This automated approach allows farmers to receive timely, region-specific, and culturally appropriate advice, enhancing productivity and sustainability [4-9].

2. LangGraph: Composing Fully Autonomous AI Agents for Agricultural Decision-Making

At the core of the AI-based crop recommendation system lies LangGraph, a composable and modular language model framework designed to automate decision-making workflows. LangGraph enables the creation of AI agents that can perform a series of tasks autonomously, without requiring human intervention once the initial process has been triggered. In the context of agriculture, this includes tasks such as retrieving weather data, analyzing it, predicting the most suitable crop to plant, and delivering actionable recommendations to the farmer—all without the need for human input during the process.

2.1. Core Features of LangGraph

1) Modular and Composable Design: LangGraph is built with modularity in mind, allowing for the easy integration of different components, or nodes, that can represent various tasks. Each node performs a specific function, and these nodes can be combined into a complete workflow. Once the system is set in motion, all nodes execute automatically. For example, after the weather data fetch node collects relevant meteorological data, the decision tree node will process that data and generate crop recommendations based on the predefined logic. The process is fully automated, ensuring that there is no need for ongoing human input after initiation.
2) State-Aware Functionality: LangGraph's state-aware functionality allows it to adjust its operations based on the specific location and environmental conditions of the farmer. By mapping geographic coordinates to specific regions, LangGraph ensures that the AI agent generates localized advice that is specific to the climate, soil conditions, and agricultural practices of the region. Once the system receives location data, it autonomously adjusts all subsequent steps in the workflow to be regionally appropriate.
3) Autonomous Multilingual Communication: The ability to communicate in multiple languages is essential for reaching a diverse group of farmers. LangGraph supports multilingual communication, allowing the AI agent to automatically translate and deliver recommendations in the farmer's native language. This process is fully automated, meaning the system will translate the crop recommendation, irrigation advice, and any other relevant information without requiring manual intervention. Once triggered, the AI agent autonomously handles all communication, ensuring that it is accessible to farmers in their preferred language.
4) Predictive Decision-Making with Decision Trees: LangGraph's integration of decision tree algorithms enables the AI agent to make automated decisions based on a variety of inputs, such as weather forecasts, soil conditions, and crop data. Once the AI agent has access to the relevant data, it automatically analyzes the environmental conditions and determines the most suitable crops to plant. The decision tree operates autonomously and without manual input, following a series of if-then rules to make transparent and interpretable decisions. This allows the system to provide farmers with accurate, data-driven recommendations, all executed automatically.
5) Seamless API Integration: LangGraph's ability to seamlessly integrate external APIs is a key feature that enables the automation of the crop recommendation process. By connecting with weather APIs such as AgroMonitoring, LangGraph automatically retrieves real-time weather data without requiring human action. Once the weather data is collected, it is processed and analyzed by the AI agent. From the data fetching to the final recommendation, every step is executed autonomously, ensuring that farmers receive timely and relevant insights.

2.2. How LangGraph Powers Fully Autonomous Crop Recommendation

1) LangGraph's ability to create fully autonomous AI agents means that once the initial process is triggered by the farmer, the system handles every subsequent task without further human input. The following is a breakdown of the process:
2) Weather Data Collection: The Weather Fetch Node autonomously queries weather APIs such as AgroMonitoring. Using the farmer's geographic location (latitude and longitude), the node retrieves live weather data, including temperature, humidity, rainfall, and other relevant parameters. The AI agent does this automatically, ensuring that the farmer does not need to manually input

weather data or perform any other actions.

3) Geolocation and Regional Mapping: The State Detection Node automatically maps the geographic coordinates to the correct region. This step ensures that the weather data is appropriately matched with the relevant agricultural practices and conditions for that region. Once the location is identified, LangGraph autonomously adjusts the recommendations to align with local farming practices, climate conditions, and crop suitability.
4) Automated Crop Decision-Making: The Crop Decision Node applies the decision tree logic to automatically predict the most suitable crops for planting based on the weather forecast and environmental conditions. Once the necessary data is collected, the system makes the decision autonomously, analyzing a range of factors like temperature, soil moisture, and humidity, and then selecting the most appropriate crops for the given conditions. The decision is made entirely by the AI agent, with no human intervention required.
5) Multilingual Advisory Generation: After the crop decision is made, LangGraph generates an automated advisory for the farmer. The Translation Node autonomously converts the advisory into the farmer's native language, using integrated translation tools such as the Google Translate API or GPT-based multilingual capabilities. Once the advisory is ready, it is automatically sent to the farmer in the appropriate language, ensuring accessibility for farmers in regions with diverse linguistic backgrounds.
6) Irrigation and Rainwater Harvesting Recommendations: Based on the weather forecast and soil conditions, the Irrigation Advisory Node and Rainwater Harvesting Node generate relevant recommendations for water management. The irrigation schedule is automatically adjusted based on the predicted rainfall, ensuring that farmers are informed about when and how much to irrigate. Similarly, if the forecast predicts heavy rainfall, the system will automatically suggest rainwater harvesting strategies to help the farmer capture and store water for future use.
7) Automated Alerts and Notifications: In addition to providing the crop recommendations, LangGraph can send automated alerts and reminders to the farmer. These notifications can include reminders for irrigation, pest management, or harvesting, all sent through the farmer's preferred communication platform. Once the initial process is triggered, these alerts and reminders are sent automatically, ensuring that farmers remain informed and on track without needing to manually monitor the system.

2.3. LangGraph's Fully Autonomous AI Agents in Action

LangGraph's fully autonomous AI agents bring several key advantages to agricultural decision-making:

1) Efficiency: By automating the entire workflow from data fetching to advisory delivery, LangGraph eliminates the need for human intervention at each step. This increases efficiency and allows farmers to receive timely recommendations without having to engage in manual data entry or decision-making.
2) Accuracy and Timeliness: The integration of real-time weather data ensures that the crop recommendations are based on the most current and accurate information available. This enables farmers to make informed decisions quickly, ensuring that they can take action before adverse weather conditions affect their crops.
3) Scalability: The composable nature of LangGraph allows it to easily scale to accommodate more data sources, models, and features as needed. For example, the system could be extended to include soil health monitoring, pest detection, or market price forecasting, all of which could be processed autonomously by the AI agents.
4) Multilingual Inclusivity: The multilingual capabilities of LangGraph ensure that the system can serve farmers in a wide range of regions and languages. This reduces barriers to understanding and ensures that all farmers, regardless of language, can benefit from the system's insights.
5) In conclusion, LangGraph allows for the creation of fully autonomous AI agents capable of managing the entire crop recommendation and advisory process, from data collection to final delivery, without human intervention. This system ensures that farmers receive timely, accurate, and region-specific recommendations, empowering them to make more informed decisions and optimize their agricultural practices. By automating the decision-making process, LangGraph represents a significant step toward a fully automated and sustainable agricultural future [10].

3. Role of Live Weather APIs

Live weather APIs, such as those provided by OpenWeatherMap, AgroMonitoring, or AccuWeather, deliver real-time and forecasted weather data essential for crop prediction. These APIs supply parameters like:

1) Current temperature
2) Humidity
3) Rainfall (historical and forecasted)
4) Wind speed
5) Soil moisture (in some cases)
6) UV index and sunlight hours

4. Integration of Weather APIs

APIs such as AgroMonitoring provide endpoints that can be queried using geographic coordinates (latitude and longi-

tude) to obtain localized weather data. Sample API parameters include:

Base URL: https://api.agromonitoring.com/agro/1.0/weather

Parameters:

1) lat: Latitude of the location
2) lon: Longitude of the location
3) appid: API key

Example Query:

https://api.agromonitoring.com/agro/1.0/weather?lat=19.076&lon=72.877&appid=YOUR_API_KEY

The API returns a JSON object containing weather forecasts, which can be integrated into crop prediction workflows.

5. The Crop Recommendation Workflow-Material and Methods

The process of determining the most appropriate crop to plant is intricate, relying on the convergence of weather data, soil conditions, crop suitability models, and local knowledge. Here, we outline a more granular breakdown of the workflow that drives the decision-making process.

Step 1: Data Preprocessing

Data preprocessing is the first step in ensuring that all input weather data is usable for prediction models. It involves several tasks to clean, normalize, and aggregate the data into a form suitable for analysis.

Normalization: Weather data, such as temperature, is often received in varying units (Celsius, Fahrenheit, etc.), and each parameter may be on different scales. Normalization standardizes these values so that they can be processed by the model effectively.

Aggregation: For accurate predictions, the system aggregates weather data over a 15-day window. This aggregated data smooths out short-term weather fluctuations and provides a stable basis for predicting trends. For instance, a 15-day rolling average of temperature and rainfall might be used to predict the optimal conditions for planting specific crops.

Validation: The aggregated weather data is compared against predefined agronomic thresholds—these thresholds reflect the ideal conditions for various crops. For example, rice requires a minimum of 100mm of rainfall and specific temperature ranges, and these thresholds help filter out unsuitable crops.

Step 2: Decision Tree Model for Crop Prediction

A decision tree model is central to predicting the best crop based on environmental variables. Decision trees are a supervised learning technique that splits the data into branches based on feature values, such as temperature, humidity, rainfall, and soil pH. These branches represent "if-then" conditions, allowing the model to predict outcomes that match a set of known conditions [11-18].

In this system, the decision tree classifier is trained on a dataset of crop requirements that considers all the relevant weather parameters. For example, the dataset could look like as shown in table 1:

Table 1. *Sample table for dataset of crop requirements.*

Crop	Temp_Min	Temp_Max	Humidity_Min	Humidity_Max	Rainfall_mm	Soil_moist_Min	Soil_moist_Max	UV_index
Rice	20	35	70	90	100-250	5.0	6.5	6-8
Wheat	12	25	50	80	30-50	6.0	7.5	4-6

Using such data, the model learns to predict which crops are most suited to different environmental conditions. The decision tree logic follows a series of conditions, such as:

If the temperature is between 25 ℃ and 30 ℃ and rainfall is moderate (100-200mm), then rice is recommended.

Step 3: Advisory Generation in LangGraph

The LangGraph framework forms the backbone of the AI agent, providing flexibility and composability for building dynamic decision trees and workflows. LangGraph is designed to allow developers to define complex workflows by creating nodes that represent specific tasks. These nodes can be linked together in a directed acyclic graph (DAG) to form the overall system logic. Each node performs a specific action in the recommendation system, contributing to the final advisory output.

6. Components of LangGraph

1) Weather Fetch Node: Queries the weather data from the AgroMonitoring API or any other weather service to retrieve up-to-date climate conditions.
2) State Detection Node: Maps the geographical coordinates (latitude and longitude) to an Indian state to ensure the advisories are region-specific. This node considers localized farming practices, crop preferences, and regional agricultural conditions.

3) Crop Decision Node: This node applies the decision tree logic and models to determine the best crop based on the weather data and region. The node takes the normalized weather data and applies the model trained on the crop dataset to determine the crop recommendation.
4) Irrigation Advisory Node: Based on the predicted rainfall and soil moisture, this node generates irrigation advisories, such as the frequency and volume of water needed for the selected crop.
5) Rainwater Harvesting Node: Suggests rainwater harvesting solutions if the forecast indicates adequate rainfall. The system considers both environmental impact and economic feasibility when making these suggestions.
6) Translation Node: After generating the final advisory, this node uses Google Translate API or GPT multilingual tools to translate the advisory into the farmer's local language, such as Hindi, Marathi, Tamil, or others.
7) LangGraph ensures that the workflow remains flexible and extensible. Additional nodes can be added to improve functionality, such as including pest control advisories, market price predictions, or even soil health recommendations.

7. Benefits of Multilingual Communication

The multilingual system offers several benefits:

1) Increased Accessibility: Farmers from different linguistic regions can access the same information.
2) Enhanced Understanding: Recommendations are delivered in a language that the farmer is comfortable with, making complex agricultural concepts easier to understand.
3) Cultural Sensitivity: Localized advice can take into account regional farming practices, dialects, and traditions, ensuring that the advisory is culturally sensitive.
4) Benefits for Farmers Using an AI-Powered Multilingual Agent System.
5) The integration of multilingual AI agents into agriculture is revolutionizing how farmers access information, receive guidance, and engage with their communities. This AI-driven framework goes far beyond basic translation—it offers intelligent, context-aware, fully automated support that is tailored to the farmer's language, culture, and local agricultural practices.

8. Multilingual Communication for Customized Farming Support

A standout feature of the system is its multilingual capability as shown in figure 1, which ensures that farmers receive advice in their native language, whether it's Hindi, Swahili, Spanish, or any other regional dialect. This significantly enhances comprehension and user engagement. The AI agents deliver personalized guidance on farming techniques, pest control, soil management, and crop care, ensuring the advice is not only understandable but also regionally and culturally relevant.

Moreover, the system supports voice-based interaction, allowing farmers who are less literate or visually impaired to communicate through speech. The agent listens, understands the query in the regional dialect, processes it, and then responds accordingly—completely eliminating the need for reading or typing. The AI also adjusts its responses based on cultural nuances, such as regional planting traditions, irrigation customs, or harvest festivals, to deliver advice that feels authentic and trusted.

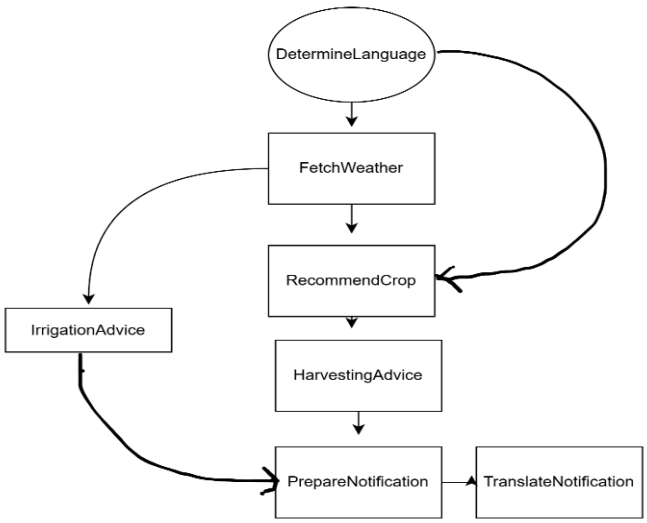

Figure 1. *LangGraph for the ai assisted farming.*

8.1. Real-Time Translation and Interpretation

AI agents serve as real-time interpreters between farmers and agricultural experts who may not speak the same language. This fosters seamless communication during consultations, village workshops, and training sessions. When agricultural experts deliver technical presentations or manuals, the system translates them into simplified and locally relevant language formats, making complex topics understandable.

Additionally, farmers participating in online forums or group discussions benefit from automatic translation of chats and posts, enabling real-time multilingual interaction. This means farmers from different regions and linguistic backgrounds can share challenges and solutions without barriers.

8.2. Localized Content Generation for Education and Awareness

The AI system also plays a vital role in creating educational content such as leaflets, blogs, infographics, and even audio or video tutorials in local languages. This material focuses on sharing best farming practices, innovative techniques, government schemes, and climate-smart agriculture.

It can also be integrated with popular messaging platforms like WhatsApp or Telegram, delivering timely information directly to farmers on channels they already use. These messages are language-specific, culturally sensitive, and often include interactive elements like quizzes or short tips.

8.3. Automated Alerts and Task Reminders

One of the most practical advantages is the system’s ability to send automated weather alerts and activity reminders in the farmer’s native language. These alerts help mitigate risks related to unexpected weather events by notifying farmers when to prepare fields, irrigate, or apply fertilizers.

Task reminders such as "time to plant maize" or "expected rainfall tomorrow" are generated using real-time weather and crop cycle data and are communicated using simple, familiar language or voice prompts.

8.4. Fostering Community and Cultural Engagement

AI agents help build stronger farming communities by enabling group discussions in local languages. Whether through text-based forums or live audio rooms, the system facilitates moderated discussions where farmers can share insights, learn from one another, and collaborate.

In addition, the AI system can generate scripts for radio shows or agricultural podcasts, helping local radio stations deliver informative programming that resonates with regional audiences. These programs play a crucial role in reaching rural communities with limited access to digital devices.

8.5. Trust Building and Inclusivity

The AI system is dialect-sensitive, ensuring that regional variations in language are respected and properly interpreted. This precision fosters trust and acceptance among users, who feel heard and understood. To overcome literacy barriers, the system can switch between text-to-speech and speech-to-text modes, allowing farmers to listen to messages or dictate responses as needed.

By tailoring interactions to individual literacy levels and linguistic preferences, the AI agent becomes a companion and guide, not just a tool—empowering farmers to make informed decisions, adopt sustainable practices, and engage with their communities confidently.

9. Results and Discussion

The introduction of LangGraph into the crop recommendation system greatly enhances both its flexibility and scalability. Farmers across India, or in any region where the system is deployed, can receive real-time advice based on localized weather data. By tailoring recommendations to the specific climatic conditions of their area, the system helps reduce the risks associated with crop failure and poor yields.

The AI system's success is also rooted in the decision tree model's transparency, which helps farmers understand why certain crops are recommended based on the weather data they provide. This transparency builds trust among users and fosters better decision-making. Additionally, by recommending appropriate crops, irrigation practices, and rainwater harvesting solutions, the system promotes sustainable agriculture practices, helping farmers optimize resources and reduce waste.

The inclusion of multilingual capabilities is a critical feature for ensuring that the advisory system serves the widest possible audience. India alone has over 22 officially recognized languages, each with various dialects. Providing information in the farmer's preferred language not only makes the system more accessible but also ensures that the knowledge is easily understood and actionable.

For example, if the system outputs a recommendation in English, the Google Translate API can translate this to Hindi:

English Output: "Based on your location and the forecasted weather, rice is the most suitable crop to plant now. Ensure irrigation every 3 days. Rainwater harvesting is highly recommended."

Hindi Output: "आपके स्थान और मौसम पूर्वानुमान के आधार पर, चावल इस समय लगाने के लिए सबसे उपयुक्त फसल है। हर 3 दिन में सिंचाई करें। वर्षा जल संचयन अत्यंत आवश्यक है।"

10. Conclusion and Future Scope

The advancement of artificial intelligence in agriculture

marks a significant turning point in how farming practices are evolving in the face of modern challenges like climate change, declining productivity, and linguistic diversity among farming communities. This research has presented a fully autonomous, intelligent, and multilingual AI agent built using the LangGraph framework, seamlessly integrated with AgroMonitoring weather APIs and decision tree algorithms. Once the process is triggered with location input, the system independently performs every subsequent step—from fetching real-time weather data to analyzing it, running predictions, generating irrigation and rainwater harvesting advisories, and finally translating the output into the farmer's local language—all without the need for human intervention.

The autonomy of this AI agent is its most transformative feature. Farmers no longer have to navigate multiple platforms, interpret complex data, or manually translate information. The AI agent acts as an all-in-one assistant that understands the environment, evaluates agricultural needs, and communicates advice in a culturally and linguistically relevant way. This automation not only saves time but also ensures consistency, reduces errors, and increases scalability for deployment in remote areas where human agricultural extension services may be sparse or non-existent.

The system's use of decision trees further enhances transparency and interpretability, which are crucial in building trust among farmers. Unlike black-box AI models, decision trees allow both developers and users to understand how a particular recommendation was reached. Furthermore, the integration of multilingual capabilities bridges the gap between cutting-edge technology and traditionally underserved farming populations, promoting inclusivity and equitable access to agricultural intelligence.

However, while the presented system shows promising results and offers immediate utility, its potential is far from exhausted.

The system could evolve into a decentralized network where farmers' feedback helps retrain the decision-making algorithms. By leveraging local insights, the AI agent becomes more robust and adaptive to region-specific needs and experiences.

In conclusion, the LangGraph-based AI agent is not merely a technological tool—it is a leap forward in democratizing agricultural intelligence, especially for those on the margins of the digital divide. With ongoing advancements and user-centered design, it holds the promise to revolutionize agriculture, making it smarter, more inclusive, and more sustainable for future generations.

Abbreviations

API	Application Programming Interface
IoT	Internet of Things
NLP	Natural Language Processing
TTS	Text-to-Speech
STT	Speech-to-Text
LLM	Large Language Model
GPS	Global Positioning System
UX	User Experience
ML	Machine Learning
UI	User Interface
DAG	Directed Acyclic Graph
OWM	Open Weather Map
NLU	Natural Language Understanding
NLG	Natural Language Generation
ICT	Information and Communication Technology

Acknowledgments

We acknowledge the assistance of ChatGPT-4.0, an advanced AI language model, for providing insights, suggestions, and support in crafting this article. Its capabilities significantly enhanced the quality and depth of the content presented.

Funding

This work received no external funding.

Conflicts of Interest

The authors declare no conflicts of interest.

Appendix

```
from langgraph.graph import StateGraph, END
from langchain.tools import tool
from langchain.chat_models import ChatOpenAI
import requests
from dotenv import load_dotenv
import os
load_dotenv()
AGRO_API_KEY = os.getenv("AGRO_API_KEY")
# Language map for Indian states
STATE_LANGUAGE_MAP = {
"Uttar Pradesh": "hi",
"Maharashtra": "mr",
"Tamil Nadu": "ta",
"West Bengal": "bn",
"Gujarat": "gu",
"Karnataka": "kn",
"Punjab": "pa",
"Telangana": "te",
# Add more mappings as needed
}
# Define the state
class FarmState(dict):
pass
# Tool: Get user's state from coordinates
```

```
@tool
def determine_local_language(state: FarmState) -> FarmState:
lat, lon = state["lat"], state["lon"]
url = f"https://nominatim.openstreetmap.org/reverse?format=json&lat={lat}&lon={lon}&zoom=5&addressdetails=1"
headers = {'User-Agent': 'farm-agent'}
res = requests.get(url, headers=headers)
if res.ok:
data = res.json()
detected_state = data.get("address", {}).get("state", None)
state["region"] = detected_state
state["language_code"] = STATE_LANGUAGE_MAP.get(detected_state, "hi")
else:
state["region"] = "Unknown"
state["language_code"] = "hi"
return state
# Tool: Fetch weather data
@tool
def fetch_weather(state: FarmState) -> FarmState:
lat, lon = state["lat"], state["lon"]
url = f"http://api.agromonitoring.com/agro/1.0/weather?lat={lat}&lon={lon}&appid={AGRO_API_KEY}"
response = requests.get(url)
if response.ok:
data = response.json()
weather_summary = {
"temp": data['main']['temp'],
"humidity": data['main']['humidity'],
"desc": data['weather'][0]['description']
}
state["weather"] = weather_summary
else:
state["error"] = "Weather API failed"
return state
# Tool: Recommend crops
@tool
def recommend_crop(state: FarmState) -> FarmState:
desc = state["weather"]["desc"]
if "rain" in desc:
state["crop_recommendation"] = "Grow rice or maize due to expected rain."
else:
state["crop_recommendation"] = "Grow millet or pulses; dry conditions expected."
return state
# Tool: Irrigation advice
@tool
def irrigation_advice(state: FarmState) -> FarmState:
desc = state["weather"]["desc"]
if "rain" in desc:
state["irrigation_advice"] = "Reduce irrigation; natural rainfall expected."
else:
state["irrigation_advice"] = "Use drip irrigation twice a week."
return state
# Tool: Rainwater harvesting
@tool
def harvesting_suggestion(state: FarmState) -> FarmState:
desc = state["weather"]["desc"]
if "rain" in desc:
state["harvesting_advice"] = "Prepare rainwater harvesting pits or tanks now."
else:
state["harvesting_advice"] = "No rain expected; maintain current water resources."
return state
# Tool: Prepare final notification
@tool
def prepare_notification(state: FarmState) -> FarmState:
notification = (
f"🌦 Weather: {state['weather']['desc']}, {state['weather']['temp']}K\n"
f"🌱 Crop Advice: {state['crop_recommendation']}\n"
f"💧 Irrigation: {state['irrigation_advice']}\n"
f"☔ Harvesting Tip: {state['harvesting_advice']}"
)
state["notification"] = notification
return state
# Tool: Translate notification
@tool
def translate_to_local_language(state: FarmState) -> FarmState:
lang_code = state.get("language_code", "hi")
llm = ChatOpenAI(model="gpt-4", temperature=0)
prompt = f"Translate the following message into the language code '{lang_code}':\n\n{state['notification']}"
translated = llm.predict(prompt)
state["notification_translated"] = translated
return state
# Build the LangGraph
from langgraph.graph import StateGraph
def build_farming_graph():
workflow = StateGraph(FarmState)
workflow.add_node("DetermineLanguage", determine_local_language)
workflow.add_node("FetchWeather", fetch_weather)
workflow.add_node("RecommendCrop", recommend_crop)
workflow.add_node("IrrigationAdvice", irrigation_advice)
workflow.add_node("HarvestingAdvice", harvesting_suggestion)
workflow.add_node("PrepareNotification", prepare_notification)
workflow.add_node("TranslateNotification", translate_to_local_language)
```

```
    workflow.set_entry_point("DetermineLanguage")
    workflow.add_edge("DetermineLanguage", "FetchWeather")
    workflow.add_edge("FetchWeather", "RecommendCrop")
    workflow.add_edge("RecommendCrop", "IrrigationAdvice")
    workflow.add_edge("IrrigationAdvice", "HarvestingAdvice")
    workflow.add_edge("HarvestingAdvice", "PrepareNotification")
    workflow.add_edge("PrepareNotification", "TranslateNotification")
    workflow.add_edge("TranslateNotification", END)
    return workflow.compile()
    # Example usage
    graph = build_farming_graph()
    result = graph.invoke({
    "lat": 18.5204, # Example: Pune
    "lon": 73.8567
    })
    print("\n✉ Final Notification (Translated):")
    print(result.get("notification_translated", "Translation failed."))
```

References

[1] LangChain & LangGraph Documentation. https://docs.langchain.com/

[2] AgroMonitoring. (n.d.). API Documentation: Weather Forecast and Analysis for Agriculture. Retrieved from https://agromonitoring.com/api

[3] AccuWeather. (n.d.). AccuWeather APIs Overview. Retrieved from https://developer.accuweather.com/apis

[4] Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). Classification and Regression Trees. Monterey, CA: Wadsworth and Brooks. https://doi.org/10.1201/978131513947

[5] Chen, Y., & Gupta, A. (2018). Weather Forecasting Applications Using APIs and Decision Tree Models. International Journal of Computer Science and Applications, 15(3), 25-34.

[6] FAO. (2011). Soil Nutrient Assessment for Sustainable Agriculture. Food and Agriculture Organization of the United Nations. Retrieved from http://www.fao.org/

[7] Singh, R., & Meena, V. (2021). AI Integration in Indian Agriculture: Opportunities and Challenges. Journal of Precision Farming, 3(2), 45-52.

[8] CustomGPT Documentation. (n.d.). API Configuration and Use Cases. Retrieved from https://customgpt.ai/docs

[9] World Bank. (2020). Climate Smart Agriculture. Retrieved from https://www.worldbank.org/en/topic/climate-smart-agriculture

[10] Pandey, N., & Kumar, P. (2020). Decision Support Systems in Agriculture: A Case Study of Weather-Driven Crop Prediction. Advances in Intelligent Systems and Computing, 4(1), 71-84.

[11] Neha Bansal, Bhawna Singla "Sustainable Green Agriculture Initiative by India" Annual International Congress on Computer Science, Oxford, United Kingdom, April 17-18, 2025.

[12] Neha Bansal, Bhawna Singla "Green Agriculture initiative by India - strengthening India Guyana Ties" Annual International Congress on Computer Science, Oxford, United Kingdom, April 17-18, 2025.

[13] Neha Bansal, Bhawna Singla "Healthcare Facilities Enhancement Using Artificial Intelligence Chatbots" Annual International Congress on Computer Science, Oxford, United Kingdom, April 17-18, 2025.

[14] Bansal, N., Singla, B. (2024). Heart Health Detector GPT Based on GPT-4o Model. *Automation, Control and Intelligent Systems*, *12*(4), 114-124. https://doi.org/10.11648/j.acis.20241204.13

[15] Singla, B. (2025). Innovating Education: The journey towards a sustainable, women- led, and digitally transformed education 5.0 via G20 summit at India. *Journal of Computer Science*, *18*(1). https://doi.org/10.15229/JCS.2025.V18I1.1483

[16] Singla, B. (2024) *Fine-Tuning FaceNet for Celebrity Face Recognition: Enhancing Real vs. Fake Image Detection on the '105_Classes_Pins' Dataset*, *Degres Journal*. Available at: https://degres.eu/volume-9-issue-12-2024/

[17] Rakesh Tayal, Arindam Bhattacharya, Bhawna Singla, Blended Education – Best Practice For Quality Improvement At Panipat Institute Of Engineering And Technology, NOVYI MIR Research Journal, Issn No:0130-7673, PAGE NO: 22-27, 2024, https://doi.org/16.10098.NMRJ.2024.V9I12.256342.38073

[18] Bhawna Singla, Soham Taneja, Rishika Garg, Preeti Nagrath, "Liver Disease Prediction using Machine Learning and Deep Learning - A Comparative Study" Accepted in Scopus Indexed Journal- Intelligent Decision System, 2021.