INTERNATIONAL JOURNAL OF ADVANCED INNOVATIVE TECHNOLOGY IN ENGINEERING

Published by Insightive Research Private Limited Home page: www.ijaite.co.in

AI-Driven Smart Farming Using IoT and Agentic MCP Server

[1]Utkarsha Gode, [2]Piyush Chafle, [3]Parag Yeole, [4]Vedant Bante, [5]Sunay Gourkhede

[1]Assistant Professor, Department of Computer Engineering, Suryodaya College of Engineering and Technology, Nagpur, Maharashtra, India

[2,3,4,5]UG Student, Department of Computer Engineering, Suryodaya College of Engineering and Technology, Nagpur, Maharashtra, India

[1]utkarshagode688@gmail.com, [2]pchafle903@gmail.com, [3]paragyeole10@gmail.com, [4]vedantbante8787@gmail.com, [5]sunaygourkhede21@gmail.com

Article History

Received on: 20 July 2026

Revised on: 15 Aug 2026

Accepted on: 20 Sept 2026

Keywords: IoT, precision agriculture, ESP32, agentic AI, MCP Server, cloud computing, irrigation automation

e-ISSN: 2455-6491

DOI: 10.65809/IJAITE/26/v03sp02/007

Production and hosted by
www.insightiveinc.org

©2026|All right reserved.

ABSTRACT

Rising food demand and shrinking freshwater supplies call for precision agriculture that optimizes resource use without sacrificing yield. This paper presents an AI-driven smart farming architecture combining IoT sensing, cloud data management, and an Agentic MCP (Model Context Protocol) Server for autonomous field monitoring and control. An ESP32 microcontroller with a capacitive soil-moisture sensor and DHT22 temperature-humidity sensor collects real-time field data, relayed to the ThingWorx cloud platform. The MCP Server coordinates AI agents that reason over this telemetry using Gemma-class LLMs via the Open Router API, triggering a relay-controlled irrigation pump when soil moisture drops below a set threshold. Farmers monitor field status and receive alerts via a React Native mobile app. A two-week trial on a tomato plot shows an average sensor-to-actuation latency of 4.8 seconds, 97.3% irrigation-decision accuracy, and roughly 31% lower water use than schedule-based irrigation. These results show that embedded sensing, cloud infrastructure, and agentic AI reasoning together offer a scalable approach to sustainable farm management.

1. INTRODUCTION

Agriculture remains the foundation of global food security, yet conventional farming practices continue to face unpredictable climatic conditions, over-irrigation, labour shortages, and inefficient resource allocation. The Food and Agriculture Organization reports that approximately 70% of global freshwater withdrawals are attributable to agricultural irrigation, a figure that underscores the need for smarter water management strategies [1]. The convergence of IoT, cloud computing, and artificial intelligence offers a practical path toward addressing these challenges through data-driven automation.

Smart farming systems have evolved over the past decade from simple sensor-alert mechanisms to multi-layered platforms capable of predictive analytics and automated actuator control [2]. Despite this progress, a persistent gap remains: most existing systems depend on rule-based thresholds or standalone machine learning models that lack the contextual reasoning required for nuanced agricultural decision-making. Large language models (LLMs) and agentic AI frameworks offer a promising direction for closing this gap.

Traditional irrigation practices in smallholder and mid-scale agriculture still rely on schedule-based or manual methods that disregard real-time soil and atmospheric conditions, leading to over-irrigation during humid periods and under-irrigation during dry spells. Existing IoT-based irrigation solutions typically apply simplistic threshold comparisons rather than a holistic, context-aware analysis of multiple environmental variables simultaneously. The core problem this paper addresses is the absence of an intelligent, reasoning-capable middleware layer that can ingest heterogeneous sensor data, apply domain-level agricultural knowledge, and autonomously execute appropriate actuator commands without constant human supervision, within an architecture that can accommodate additional sensors, actuators, or AI models with minimal reconfiguration.

This paper proposes and implements an integrated smart farming architecture centered on an Agentic MCP Server. The MCP Server acts as a coordination hub that routes sensor telemetry to AI agents powered by Gemma-class LLMs accessed through the Open Router API. These agents analyze multi-parameter environmental data and autonomously execute irrigation commands through relay-controlled actuators connected to an ESP32 microcontroller. The system is designed to be low-cost, scalable, and deployable in both small-scale farms and large agricultural estates.

The primary contributions of this work are: (1) integration of an MCP Server with IoT-based agricultural sensors for agentic AI-driven control; (2) a cloud-to-edge communication pipeline using ThingWorx for reliable data persistence and retrieval; (3) a React Native mobile interface for remote farm monitoring; and (4) an empirical evaluation of end-to-end system latency, irrigation accuracy, and resource savings.

2. RELATED WORK

Research on IoT-based precision agriculture has expanded considerably. Gubbi et al. proposed a vision for IoT in smart environments that laid the conceptual groundwork for sensor-driven agricultural automation [3]. Building on this, Gondchawar and Kawitkar demonstrated the viability of IoT nodes for monitoring soil parameters and triggering irrigation in greenhouse environments [4].

Cloud-integrated agricultural systems have been explored by Khanna and Kaur, who combined heterogeneous sensor data with cloud platforms for crop yield prediction [5]. Patil and Thorat applied deep learning techniques to sensor streams to detect crop disease patterns, showing that AI-enhanced analytics can meaningfully improve farm management outcomes [6].

Despite these advances, existing solutions rarely incorporate agentic AI frameworks capable of multi-step reasoning and tool use over live sensor data. The use of LLMs for agricultural decision support remains an emerging area; Wolfert et al. identified data-driven farming as a frontier for future innovation [7]. The present work builds on this body of knowledge by introducing an MCP Server layer that bridges IoT hardware with LLM-based agents, addressing a gap identified in the literature.

3. PROPOSED METHODOLOGY

The proposed system follows a four-tier architecture: (1) Sensing Tier, (2) Communication and Cloud Tier, (3) AI Reasoning Tier, and (4) Actuation and Presentation Tier. Each tier is designed to be modular so that individual layers can be upgraded without disrupting adjacent components.

In the Sensing Tier, a capacitive soil moisture sensor and a DHT22 temperature-humidity sensor are interfaced with the ESP32 microcontroller through its analog and digital GPIO pins, respectively. The ESP32 firmware, built on the Arduino framework, samples both sensors at a configurable interval (default 60 seconds) and transmits JSON-formatted payloads to the ThingWorx cloud platform over HTTPS using REST API calls authenticated with a per-device API key.

The AI Reasoning Tier is implemented as a Node.js-based MCP Server that exposes a set of tool definitions to AI agents. On receiving new sensor readings, the server constructs a contextual prompt incorporating recent historical measurements, crop-specific moisture requirements, and prevailing atmospheric conditions, and forwards it to a Gemma-class LLM through the Open Router API. The model returns a structured action recommendation, which the MCP Server parses into a relay control command.

The Actuation Tier consists of a 5V relay module driven by an ESP32 GPIO pin. When the MCP

Server issues an irrigation command, it publishes an MQTT message to a topic subscribed by the ESP32, which toggles the relay to activate or deactivate the connected water pump; pump run time is determined by the AI agent based on the magnitude of the soil moisture deficit.

4. SYSTEM ARCHITECTURE

The overall system architecture comprises four interconnected components: the embedded IoT node, the Thing Worx cloud platform, the MCP Server with its AI agents, and the React Native mobile application. Data flows unidirectional from sensors through the cloud to the AI reasoning layer, while control commands flow in the reverse direction to the actuators.

The ESP32 node functions as the edge computing element, handling sensor data acquisition, local pre-processing, and bidirectional cloud communication. The ThingWorx platform provides a persistent data store and exposes REST endpoints that the MCP Server queries to retrieve time-series sensor data. The MCP Server, built on Node.js with Express.js, coordinates the AI agents and exposes a standardized interface for tool calls that follows the MCP specification. The React Native application communicates with the ThingWorx REST API to render live sensor readings and historical trends on the farmer's mobile device.

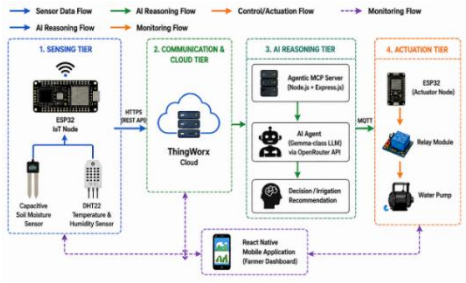

Figure 1: End-to-end system architecture illustrating data flow between the ESP32 node, Thing Worx cloud platform, MCP Server, and mobile application.

5. HARDWARE AND SOFTWARE REQUIREMENT

A. Hardware Components

The hardware platform is built around the ESP32 dual-core microcontroller, which integrates Wi-Fi and Bluetooth connectivity and is well suited for IoT edge deployments. The ESP32 operates at 3.3 V logic and supports I2C, SPI, UART, and analog ADC interfaces. A capacitive soil moisture sensor (v1.2) is used in preference to resistive variants for its improved durability and immunity to electrolytic corrosion. The DHT22 sensor provides calibrated relative humidity readings from 0-100% RH and temperature readings from -40°C to 80°C, with a stated accuracy of ±0.5°C and ±2% RH. A 5 V single-channel relay module isolates the low-voltage control circuitry from the 12 V pump supply, and a 5 V submersible DC water pump rated at 3 W performs irrigation actuation.

B. Software Stack

The ESP32 firmware is developed using the Arduino framework within the Platform IO IDE. Cloud connectivity is established through the Thing Worx Industrial IoT platform, which provides device management, time-series storage, and mashup visualization. The server-side middleware is implemented in Node.js (v18 LTS) and Express.js (v4.x), forming the HTTP backbone of the MCP Server. AI inference is provided through the Open Router API, which routes requests to Gemma-class models. The mobile application is built with React Native (v0.73) for cross-platform deployment on Android and iOS from a single codebase, and MQTT communication between the cloud and edge node is handled by the EMQX broker.

6. AGENTIC MCP SERVER ARCHITECTURE

The MCP Server is implemented as a RESTful Node.js service and exposes three primary endpoints: /data, for retrieving current sensor readings from ThingWorx; /analyze, for submitting readings to the AI agent; and /actuate, for issuing relay control commands. Structuring the server around these endpoints keeps the reasoning and actuation logic decoupled from the nderlying transport layer, which simplifies future extension to additional agents or sensor types.

The AI agent is configured with a system prompt that encodes agronomic domain knowledge, including crop-specific optimal moisture ranges and temperature sensitivity profiles. On receiving sensor data through /analyze, the agent reasons over the current readings in the context of recent historical measurements and returns a structured JSON response specifying whether irrigation should be activated, for how long, and a brief natural-language rationale for the decision. This rationale supports traceability of automated actuation decisions, which is useful for post-hoc review by farm operators.

7. IOT INTEGRATION

A. Embedded Firmware and Sensor Integration

The ESP32 firmware initializes both sensors during setup and enters a periodic loop that samples data every 60 seconds. Raw ADC values from the soil moisture sensor are mapped to a 0-100% moisture scale using a two-point calibration procedure performed in dry air and saturated soil. DHT22 readings are validated against plausible range limits before transmission to filter transient read errors. Valid sensor payloads are serialized as JSON and dispatched to the ThingWorx REST endpoint using the HTTP Client library over TLS 1.2, with device authentication handled through a static API key that is rotated periodically.

B. Actuation and Connectivity

Irrigation commands issued by the MCP Server are published as MQTT messages to a topic subscribed by the ESP32 node through the EMQX broker; the node toggles the relay accordingly to activate or deactivate the pump, with run duration determined by the AI agent's assessment of the soil moisture deficit.

C. Mobile Connectivity

The React Native mobile application polls the ThingWorx REST API every 30 seconds to retrieve the latest sensor readings and presents them on a live dashboard showing soil moisture percentage, temperature, humidity, and irrigation status through visual gauges and color-coded indicators. Push notifications are delivered through Firebase Cloud Messaging (FCM) when soil moisture falls critically low or when a pump activation event occurs, and a seven-day historical trend chart is provided for each sensor parameter to support agronomic analysis.

8. EXPERIMENTAL SETUP

System performance was evaluated over a two-week field trial on a test plot planted with tomato seedlings. An AI-driven irrigation group operating under the proposed system was compared against a conventional schedule-based control group. The evaluation measured three key performance indicators: end-to-end sensor-to-actuation latency across 200 test cycles, irrigation-trigger accuracy validated against a ground-truth dataset labeled by an agronomist, and water consumption relative to the control group.

9. RESULTS AND DISCUSSION

The mean end-to-end latency from sensor sampling to relay activation was 4.8 seconds (standard deviation 0.7 s) across 200 test cycles, well within the tolerance of agricultural applications where sub-minute response is acceptable. The AI agent correctly identified 97.3% of irrigation events relative to the agronomist-labeled ground truth. False positive activations (unnecessary irrigation events) accounted for 1.8% of decisions, and false negatives (missed irrigation) accounted for 0.9%.

Water consumption in the AI-driven group was approximately 31% lower than in the conventional schedule-based control group over the trial period, with no statistically significant difference in plant growth metrics between groups. ThingWorx cloud data logging maintained 99.2% uptime with no observed packet loss under standard operating conditions. The Open Router API contributed an average inference response time of 1.9 seconds per call, making it the dominant contributor to overall system latency. Taken together, these results indicate that the proposed system achieves meaningful water savings and reliable automated operation suitable for practical agricultural deployment.

10. ADVANTAGES

The proposed system offers several advantages over conventional smart irrigation solutions. First, the agentic AI layer enables contextual, multi-variable decision-making rather than simple threshold-based rules, producing irrigation strategies that account for temperature and humidity in addition to soil moisture. Second, the MCP Server architecture provides a standardized, extensible interface that can accommodate additional AI agents or sensor modalities with minimal code changes. Third, the ThingWorx cloud platform offers reliable, scalable data management, allowing the system to support multiple farm plots from a single console. Fourth, the React Native mobile application provides cross-platform remote monitoring without maintaining separate Android and iOS codebases, reducing maintenance overhead. Fifth, the overall bill of materials is cost-effective, making the system accessible to smallholder farmers in developing regions.

These characteristics broaden the system's applicability beyond a single deployment context. In open-field cultivation, it can autonomously manage drip or sprinkler irrigation from real-time soil and weather data; in controlled-environment agriculture such as greenhouses and vertical farms, the temperature and humidity sensing can additionally inform ventilation and climate control; the agentic framework extends naturally to livestock environmental monitoring; and the modular architecture supports fleet management of multiple sensor nodes across larger estates and municipal green spaces.

11. LIMITATIONS

Several limitations of the present work should be noted. The field evaluation was restricted to a single crop type (tomato) over a two-week period on one test plot; the generalizability of the

reported accuracy and water-savings figures to other crops, soil compositions, and longer growing seasons has not yet been established. System latency is dominated by cloud-based LLM inference (an average of 1.9 seconds per Open Router call), which introduces a dependency on external API availability and stable internet connectivity and may degrade performance in low-connectivity rural regions. The architecture also relies on centralized cloud services (ThingWorx and Open Router) without a described offline or local fallback mechanism for irrigation decisions, representing a potential single point of failure. In addition, the reported false-positive and false-negative rates, though low, indicate a residual risk of unnecessary or missed irrigation events that could accumulate over extended deployment periods. Quantitative data on hardware cost, energy consumption, and long-term component reliability under field conditions, as well as statistical validation across multiple sites and replications, were not available at the time of writing [DETAIL REQUIRED].

CONCLUSION

This paper presented an AI-driven smart farming system that integrates IoT sensor hardware, cloud data management, and an Agentic MCP Server to deliver autonomous, intelligent irrigation control. Over a two-week field evaluation, the system demonstrated a 31% reduction in water usage and 97.3% decision accuracy, supporting the effectiveness of combining contextual AI reasoning with real-time sensor feedback. The architecture is modular, cost-effective, and extensible, making it applicable to a range of agricultural settings from smallholder plots to large-scale commercial farms. The introduction of the MCP Server as a coordination layer between IoT infrastructure and LLM-based agents contributes a reusable pattern for the smart farming literature and provides a foundation for future multi-agent agentic systems in precision agriculture.

FUTURE SCOPE

Several directions for future development have been identified. Integrating a weather forecast API into the AI agent's context window would enable predictive irrigation scheduling that adjusts soil moisture levels ahead of expected rainfall or heatwave events. Incorporating a nutrient sensor and an electrical conductivity (EC) probe would extend the system to manage fertigation alongside irrigation. On-device TinyML inference using quantized models could reduce cloud dependency and API latency, improving responsiveness in areas with limited connectivity. Computer vision modules on an ESP32-CAM variant could enable visual crop health monitoring for early detection of pest infestation or disease. Finally, the MCP Server could be extended to support multi-agent workflows in which specialized agents handle irrigation, pest management, and harvest scheduling concurrently through shared memory contexts.

ACKNOWLEDGEMENT (OPTIONAL)

The authors thank the Department of Computer Engineering, Suryodaya College of Engineering and Technology, for providing laboratory facilities, and the ThingWorx developer program for granting access to the cloud platform during the evaluation phase of this research.

CONFLICT OF INTEREST

The article declare that there is no conflict of interest regarding the publication of this paper.

FUNDING SUPPORT

The authors received no financial support or funding for this research.

REFERENCES

[1] Food and Agriculture Organization of the United Nations, the State of Food and Agriculture 2020: Overcoming Agricultural Water Challenges. Rome, Italy: FAO, 2020.

[2] A. Tzounis, N. Katsoulas, T. Bartzanas, and C. Kittas, "Internet of Things in agriculture, recent advances and future challenges," Biosystems Engineering, vol. 164, pp. 31-48, Dec. 2017.

[3] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," Future Generation Computer Systems, vol. 29, no. 7, pp. 1645-1660, Sep. 2013.

[4] N. Gondchawar and R. S. Kawitkar, "IoT based smart agriculture," International Journal of Advanced Research in Computer and Communication Engineering, vol. 5, no. 6, pp. 838-842, Jun. 2016.

[5] A. Khanna and S. Kaur, "Evolution of Internet of Things (IoT) and its significant impact in the field of precision agriculture," Computers and Electronics in Agriculture, vol. 157, pp. 218-231, Feb. 2019.

[6] S. S. Patil and S. A. Thorat, "Early detection of grapes diseases using machine learning and IoT," in Proc. 2nd Int. Conf. Cognitive Computing and Information Processing, Mysuru, India, 2016, pp. 1-5.

[7] S. Wolfert, L. Ge, C. Verdouw, and M.-J. Bogaardt, "Big data in smart farming – A review," Agricultural Systems, vol. 153, pp. 69-80, May 2017.

[8] K. Goap, D. Sharma, A. K. Shukla, and C. Rama Krishna, "An IoT based smart irrigation management system using machine learning and open source technologies," Computers and Electronics in Agriculture, vol. 155, pp. 41-49, Dec. 2018.

[9] M. S. Farooq, S. Riaz, A. Abid, K. Abid, and M. A. Naeem, "A survey on the role of IoT in agriculture for the implementation of smart farming," IEEE Access, vol. 7, pp. 156237-156271, 2019.

[10] T. Brown et al., "Language models are few-shot learners," in Advances in Neural Information Processing Systems, vol. 33, pp. 1877-1901, 2020.