International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Survey on Backend Frameworks, Secure Data Handling, and API Integration for IoT-Enabled Smart Agriculture Systems

Dr. Sumathy Kingslin[1], K. Vaishnavi[2]

[1]*Associate Professor,* [2]*Research Scholar, PG Department of Computer Science, Quaid-E-Millath Government College for Women, Chennai – 02, India*

***Abstract*-- The growing adoption of IoT-based smart agriculture systems requires robust backend architectures capable of securely managing continuous sensor data and delivering real-time insights to end users. This study presents a concise survey of existing backend frameworks, secure data-handling techniques, and API mechanisms used in modern agricultural IoT platforms. Literature shows a strong preference for lightweight backend technologies such as Flask due to their compatibility with Python-based AI models, while MongoDB remains widely used for storing unstructured sensor datasets. Common security practices include AES encryption for protecting sensitive farm information and JWT-based authentication for implementing stateless, multi-role access control. The survey also identifies persistent gaps, including limited end-to-end security, fragmented system designs, and insufficient support for young farmers. The findings highlight the need for an integrated, secure, and scalable backend layer—addressed in the proposed Phase 2 architecture—to ensure reliable IoT data processing and provide a strong foundation for subsequent blockchain-enabled agricultural services.**

***Keywords*-- IoT-based Smart Agriculture, Backend Frameworks, Flask API, Secure Data Handling, AES Encryption, JWT Authentication, MongoDB Storage, RESTful Services, Role-Based Access Control, Real-Time Sensor Data.**

I. INTRODUCTION

The rapid expansion of IoT-enabled smart agriculture has transformed traditional farming by enabling continuous monitoring of soil conditions, crop health, and environmental parameters. While sensor networks generate valuable real-time data, the effectiveness of these systems depends on a reliable backend capable of processing, storing, and securing the collected information. Phase 2 of modern smart farming architectures focuses on building this essential backend layer, which acts as the bridge between field-level IoT devices, AI-based analytics, and user-facing web or mobile platforms.

Existing agricultural systems often struggle with challenges such as inconsistent sensor data, insecure communication channels, and limited mechanisms for user authentication. To overcome these issues, researchers emphasize the use of lightweight backend frameworks, secure API design, efficient database selection, and strong encryption methods. Technologies like Flask provide a flexible and AI-compatible API server, while MongoDB supports high-speed storage of heterogeneous sensor values. Meanwhile, AES encryption and JWT authentication ensure data confidentiality and controlled access for farmers, buyers, and administrators.

This article presents a focused survey of backend and security approaches commonly used in IoT-based agricultural platforms. By analysing current practices and identifying existing gaps, the study highlights the need for a unified, secure, and scalable backend architecture—forming the foundation for the proposed Phase 2 implementation in this project.

II. REVIEW OF LITERATURE

Recent work on IoT-enabled smart agriculture emphasizes lightweight, AI-friendly backend stacks, robust storage for high-frequency sensor streams, and pragmatic security measures to protect farm data. Several prototype and survey papers report that Python-based microframeworks (notably Flask) are commonly adopted in agricultural research because they integrate directly with machine-learning pipelines and enable rapid prototyping of REST APIs for sensor ingestion and model inference.

For time-series sensor data, the literature compares document stores (e.g., MongoDB) and specialized time-series databases (e.g., InfluxDB/TimescaleDB). Many studies conclude MongoDB is favored in prototypes because its JSON document model matches the structure of IoT payloads and simplifies development, while time-series DBs can offer better compression and query performance for large-scale telemetry—leading authors to recommend a hybrid approach depending on workload characteristics.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Security is a central theme. Symmetric ciphers such as AES (commonly AES-128) remain the default for on-device and backend data protection due to hardware support and acceptable performance on constrained devices; however, recent analyses highlight implementation caveats and resource trade-offs on low-power microcontrollers, prompting research into lightweight alternatives or careful AES parameter selection.

Authentication and authorization practices for IoT systems have been widely surveyed. JWT (JSON Web Token) is frequently used in RESTful IoT backends and for securing API access because of its statelessness and easy integration with web/mobile clients, though research also warns about secure token handling, rotation, and revocation strategies. Broader surveys of IoT authentication/authorization recommend layered approaches (device identity, transport security, token management, and RBAC/ABAC for fine-grained access control).

API design patterns in the agricultural IoT literature favor REST for device-to-server communication, often with validation middleware (Pydantic/Marshmallow) and logging to detect faulty sensors. Several applied systems add message-brokers (MQTT) or WebSocket layers for real-time dashboards while keeping REST endpoints for CRUD, authentication, and administrative tasks.

Across the surveyed sources, recurring gaps appear: (1) few end-to-end architectures integrate lightweight backend, in-server ML inference, robust storage, and blockchain traceability in a single, evaluated system; (2) many prototypes secure communication channels (TLS/MQTT) but do not apply encryption to persisted data or implement token revocation; (3) user-centered design for youth adoption (simplified dashboards, role-specific UX) is underexplored. These gaps motivate Phase 2 choices such as Flask + MongoDB + AES + JWT combined with clear RBAC policies.

III. Overview Of Backend Frameworks

Modern IoT–agriculture backends must handle high-frequency sensor writes, API access for web/mobile dashboards, integration with AI inference, and secure user/device authentication. Framework choice balances development speed, AI compatibility, real-time support, and production scalability.

Short comparison of common frameworks:

- Flask (Python) — Minimal, easy to prototype, excellent for embedding Python ML models directly. Best for research prototypes and Phase-2 implementations where in-server inference and rapid development matter. Requires adding libraries for validation, async handling, and large-scale production concerns.
- FastAPI (Python) — Modern, fast (ASGI), built-in validation (Pydantic), automatic OpenAPI docs, and great async support. Strong candidate when you want both Python/ML compatibility and higher throughput than Flask without much extra boilerplate.
- Django (Python) — Batteries-included (ORM, admin, auth). Good for large, data-centric applications with complex business logic (user management, transactions). Heavier to set up; overkill for simple sensor ingestion but useful if you need robust admin features quickly.
- Node.js (Express / NestJS) — Excellent for real-time streaming, WebSocket/MQTT bridging, and high concurrency. Use when you need event-driven pipelines or prefer JavaScript stack for frontend/backend parity. Less direct ML integration (requires model-serving or microservice approach).
- Spring Boot (Java) — Enterprise-grade, strong for high-throughput, strongly typed systems with complex transactions. More heavyweight; chosen when long-term maintainability, strong typing, and corporate deployment standards matter.

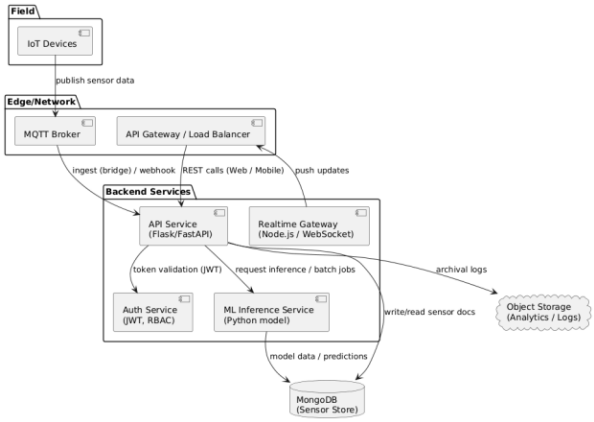

FIG 1: BACKEND ARCHITECTURE FOR IoT-ENABLED SMART AGRICULTURE SYSTEM

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

Figure 1 illustrates the overall backend architecture used in Phase 2 of the smart agriculture system. IoT devices deployed in the field send sensor data to an MQTT broker, which acts as the initial communication layer. The API service (implemented using Flask or FastAPI) receives this data, performs validation, and forwards it to the MongoDB database for secure storage. The backend also interacts with a dedicated authentication service that manages JWT-based identity verification and role-based access control. An ML inference module provides real-time predictions to the API when required. A real-time gateway (such as a Node.js WebSocket service) pushes live updates to the web and mobile dashboards. This architecture ensures efficient data ingestion, strong security, AI integration, and seamless communication between devices, backend services, and user interfaces.

IV. Secure Data Handling Methods

Secure data handling is essential in IoT-enabled smart agriculture systems because sensor values, user credentials, and marketplace transactions must remain protected throughout their lifecycle. The backend must ensure confidentiality, integrity, authenticity, and controlled access while processing high-frequency sensor data from distributed farm environments.

The first layer of protection involves securing data-in-transit. Agricultural IoT devices often transmit readings over constrained networks, making them vulnerable to interception. To mitigate this, systems commonly employ TLS/HTTPS encryption or device-to-broker secure channels. For additional protection, sensitive fields such as humidity, soil data, or fertilizer information can be encrypted at the device level using AES symmetric encryption, which provides strong security with low computational overhead suitable for IoT microcontrollers.

Once the data reaches the backend, data-at-rest security becomes crucial. Databases such as MongoDB store large volumes of JSON-based sensor records, making structured encryption and access control essential. Sensitive information is encrypted before storage, and role-based permissions restrict who can view or modify specific datasets. Database-level authentication and IP-based access rules further reduce unauthorized access risks.

Authentication and authorization mechanisms form another core component of secure handling. JWT (JSON Web Token) is widely used to validate every request coming from the front-end or mobile apps. Since JWTs are stateless, the backend efficiently verifies user identity without frequent database lookups.

Combined with role-based access control, the system ensures that farmers, buyers, administrators, and youth users each access only the features they are authorized to use.

Finally, robust input validation and error-handling procedures protect the system from malformed packets, sensor failures, and API abuse. Validation layers prevent SQL/NoSQL injection attacks, filter out noisy or corrupted sensor values, and maintain the overall integrity of the data pipeline.

Together, these security measures create a trustworthy and resilient backend capable of protecting sensitive agricultural information while supporting real-time decision-making and future blockchain-based traceability.

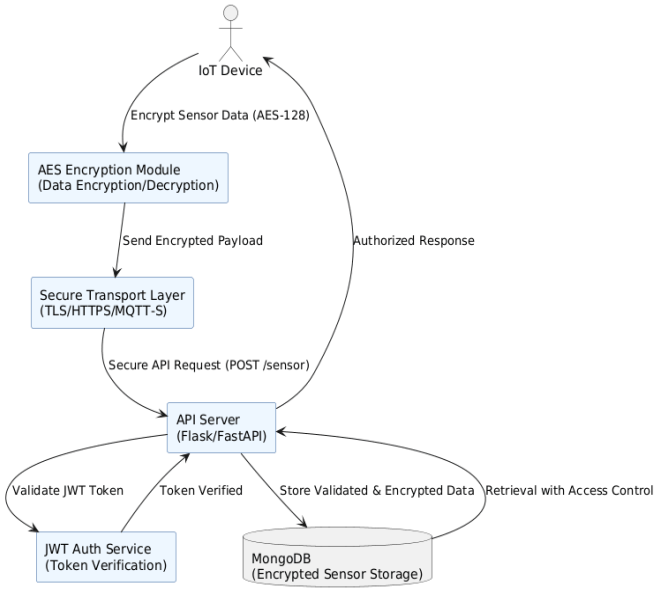

FIG 2: SECURE DATA FLOW IN THE IOT-ENABLED SMART AGRICULTURE BACKEND

Figure 2 illustrates how sensor data moves securely through the backend architecture. IoT devices encrypt the collected values using lightweight AES before transmitting them to the API server over secure channels. Upon receiving the request, the backend validates the JSON payload and verifies the user or device identity using JWT-based authentication. The validated data is then stored securely in the MongoDB database with role-based permissions controlling access. Finally, the backend returns responses only after ensuring proper authorization, maintaining confidentiality and integrity throughout the entire flow.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

1. API Integration Approaches

In IoT-enabled smart agriculture systems, the backend must provide reliable and efficient APIs to support communication between field devices, AI inference modules, and web or mobile dashboards. API design ensures smooth data ingestion, validation, secure access, and retrieval of processed results.

RESTful APIs

Most research and practical implementations favor **REST APIs** due to their simplicity, scalability, and compatibility with both web and mobile platforms. REST endpoints typically support **CRUD operations** for sensor data, user management, and transaction records. Common practices include:

- Using JSON as the standard data format for sensor payloads
- Validating incoming requests with schemas (e.g., Pydantic or Marshmallow)
- Returning structured responses with proper HTTP status codes
- Logging requests and errors for monitoring and troubleshooting

Real-Time Data APIs

Some systems require real-time streaming of sensor updates to dashboards. For these scenarios:

- **WebSockets** or **MQTT bridging services** complement REST APIs for low-latency updates
- API endpoints manage authentication and filtering, while the real-time gateway pushes data live to clients

Security in API Integration

All API endpoints enforce **JWT authentication** and **role-based access control**. Requests from mobile apps, web dashboards, or edge devices are validated for:

- Token integrity
- Expiry and revocation
- Permissions for specific operations (read/write/admin)

Together, REST endpoints for CRUD operations and real-time API channels form a robust interface for the backend to manage high-frequency IoT data while maintaining security and integrity.

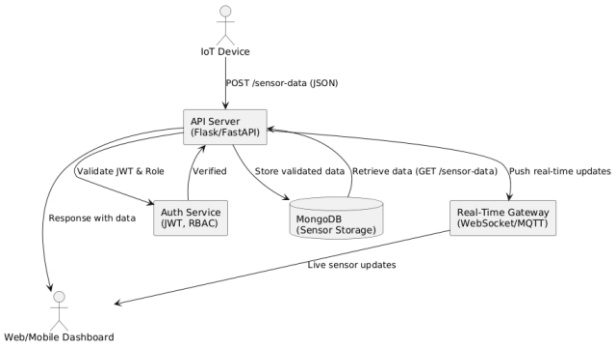

FIG 3: API INTEGRATION FLOW FOR IoT SMART AGRICULTURE BACKEND

Figure 3 depicts the API integration approach in the Phase-2 backend. IoT devices send sensor data to REST endpoints on the API server, which validates requests using JWT authentication and RBAC. The backend stores validated data in MongoDB and returns responses to web or mobile clients. For live dashboards, a real-time gateway (WebSocket or MQTT bridge) pushes sensor updates to subscribed clients. This combination of REST and real-time APIs ensures efficient, secure, and scalable communication across all system layers.

V. Identified Research Gaps

Despite significant progress in IoT-enabled smart agriculture, the surveyed literature highlights several persistent gaps that Phase 2 aims to address:

1. *Lack of Integrated Backend Architectures* – Many existing systems focus either on IoT data acquisition, AI inference, or blockchain traceability independently. Very few studies implement a fully integrated backend that combines sensor ingestion, secure storage, real-time analytics, and AI prediction in a single pipeline.
2. *Incomplete Security Measures* – While TLS or MQTT-secured channels are common, many systems do not encrypt data at rest or implement robust token management. Role-based access control and end-to-end encryption are often overlooked, leaving sensitive agricultural data partially exposed.
3. *Limited Real-Time Data Support* – Several prototypes rely solely on REST APIs for data ingestion, which may introduce latency and fail to meet real-time monitoring requirements for critical parameters like soil moisture, temperature, or crop health.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 15, Issue 02, February 2026)

4. *Insufficient User-Centric Design* – Research rarely considers the usability needs of different stakeholders such as young farmers, buyers, and administrators. Dashboards and APIs are often complex, limiting adoption among youth and non-technical users.
5. *Scalability and Extensibility Challenges* – Many backend frameworks are evaluated only on small-scale deployments. Scaling to multiple farms, thousands of sensors, and concurrent users remains underexplored, limiting real-world applicability.
6. *Limited Integration with AI and Future Blockchain Modules* – Backend systems frequently store raw sensor data but do not directly integrate ML inference pipelines or prepare data for blockchain-enabled smart contracts, reducing automation and traceability capabilities.

VI. Relevance To The Proposed System

The proposed backend design directly addresses the gaps identified in the surveyed literature. By integrating a lightweight framework (Flask/FastAPI) with MongoDB for flexible sensor storage, the system ensures efficient ingestion and storage of high-frequency IoT data. AES encryption secures sensitive sensor information both in transit and at rest, while JWT-based authentication combined with role-based access control (RBAC) enforces strict user permissions for farmers, buyers, administrators, and youth participants.

RESTful APIs provide reliable endpoints for CRUD operations, whereas a real-time gateway (WebSocket/MQTT) ensures low-latency monitoring for dashboards. Input validation and error-handling layers guarantee that noisy or corrupted sensor packets do not compromise system integrity. The backend is also designed to support seamless integration with AI inference modules and future blockchain-based traceability layers, ensuring end-to-end automation, security, and transparency.

In summary, the proposed system combines secure data handling, robust API integration, real-time support, and scalable backend architecture to create a reliable platform that meets both research and practical requirements for IoT-enabled smart agriculture.

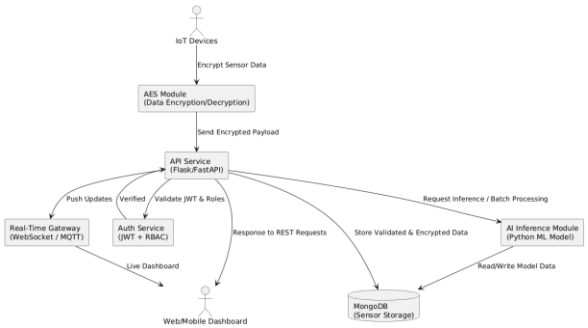

FIG 4: PROPOSED BACKEND ARCHITECTURE INTEGRATING SECURITY, API, AND REAL-TIME PROCESSING

VII. Conclusion

The survey highlights that modern IoT-enabled smart agriculture systems rely heavily on robust backend architectures capable of managing large volumes of heterogeneous sensor data while ensuring security, scalability, and real-time responsiveness. Existing studies demonstrate significant progress in individual aspects such as IoT data acquisition, cloud communication, or machine learning integration, but they often lack a unified framework that combines secure data handling, efficient API design, and adaptable backend technologies.

The proposed system architecture in Phase 2 addresses these limitations by integrating lightweight backend frameworks (Flask/FastAPI), AES-based encryption for secure data storage and transmission, JWT authentication with role-based access control, and flexible API mechanisms supporting both REST and real-time communication. This unified approach ensures the confidentiality, integrity, and availability of agricultural data while enabling seamless communication between IoT devices, AI models, user dashboards, and future blockchain modules.

Overall, the survey reinforces the relevance and necessity of a secure, scalable, and extensible backend foundation for smart agriculture applications. The insights derived from this study directly support the design choices of the proposed system and lay a strong groundwork for Phase 3, where blockchain-based traceability and smart contract automation will further enhance transparency, farmer empowerment, and trust in agri-commerce.

REFERENCES

[1] A. D. Mishra, "Farmingo: A MERN and ML-Integrated Platform for Smart, ...," *Int. J. Adv. Eng. Res. Sci.*, 2025. journals.latticescipub.com

[2] "InfluxDB vs. MongoDB — A comparison for time series workloads," InfluxData (technical paper). 3.1 years ago. InfluxData

[3] M. Has, "Efficient Data Management in Agricultural IoT," *PMCID: PMC11174974*, 2024. PMC

[4] P. Arpaia, "Problems of the advanced encryption standard in ...," *Comput. Secur.* (article discussing AES on IoT MCUs), 2020. ScienceDirect

[5] K. Shingala, "JSON Web Token (JWT) based client authentication in Message Queuing Telemetry Transport (MQTT)," arXiv, 2019. arXiv

[6] M. El-hajj et al., "A Survey of Internet of Things (IoT) Authentication Schemes," *J. Netw. Comput. Appl.*, 2019. PMC

[7] J. P. Díaz et al., "Authorization models for IoT environments: A survey," *Comput. Commun.*, 2025. ScienceDirect

[8] "Smart Secure with IoT: Agriculture Data based on Farm ...," Zenodo preprint (AES extension in agriculture), 2019–2020.