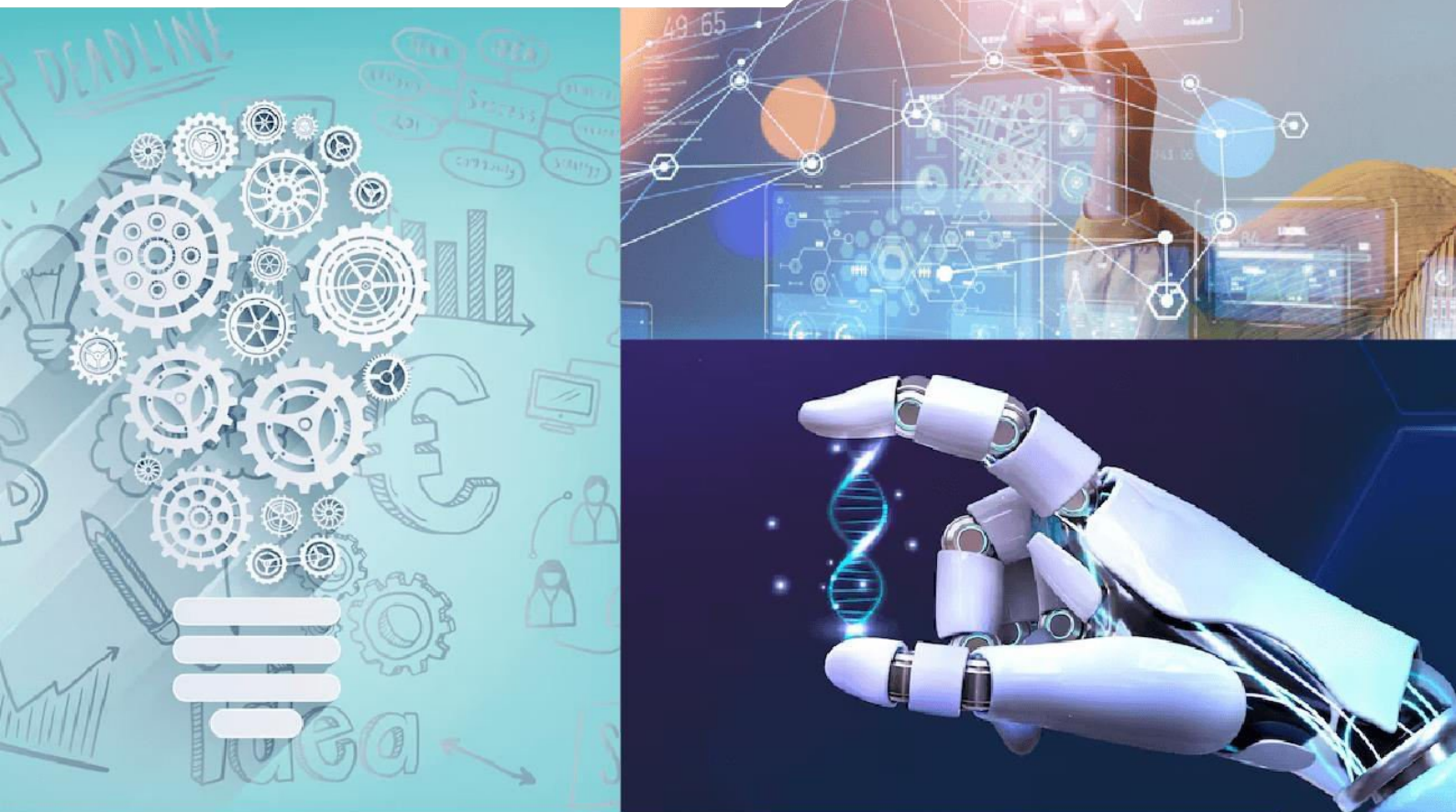




International Journal of Advanced Research in Education and Technology (IJARETY)

Volume 13, Issue 2, March - April 2026

Impact Factor: 8.152



INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA



Organic Supply Chain Traceability Using Blockchain Technology for Sustainable Agri-Business

Chayarani J, Gnaneswari Gnanaguru

Master of Computer Applications (MCA), CMR Institute of Technology, Bengaluru, India

Associate Professor, Department of Computer Applications, CMR Institute of Technology, Bengaluru, India

ABSTRACT: Verifying the authenticity of organic food products remains a persistent challenge in modern agricultural commerce. Conventional tracking mechanisms—whether centralised databases or paper-based registers—are inherently fragile: they rely on a single custodian of truth and cannot guarantee that records remain unaltered after creation. This paper describes the design, construction, and evaluation of a *Blockchain-Based Organic Supply Chain Traceability System* tailored for sustainable agri-business operations. The system records every custody transfer of an organic product—starting at the farm and ending at the point of consumer purchase—as an immutable transaction on an Ethereum blockchain. A unique Batch ID anchors each product throughout its journey. At the retail stage, a scannable QR code is affixed to the product; any consumer can retrieve the full provenance record by scanning the code, without requiring an account or prior registration. The implementation combines Ethereum smart contracts (Solidity), a React.js front-end, a Node.js/Express.js back-end, MongoDB for structured querying, and Web3.js for blockchain communication. Systematic functional testing confirmed that every workflow—from product registration through multi-stage updates to consumer verification—operates correctly and that the on-chain records resist post-registration modification. The results establish a practical, deployable framework for restoring consumer confidence in organic labelling.

KEYWORDS: Blockchain, Ethereum, Smart Contracts, Organic Supply Chain, Traceability, QR Code, Web3.js, MongoDB, Agri- Business, Food Safety, Decentralised Ledger

I. INTRODUCTION

Consumer interest in organically produced food has risen markedly over the past decade. Awareness of the health risks associated with pesticide residues, combined with a growing preference for environmentally responsible farming practices, has expanded the organic market at rates that outpace many conventional food segments [7]. Yet this growth carries an inherent tension: as the financial premium attached to organic certification increases, so does the incentive for dishonest actors to misrepresent non-organic produce as certified organic. The resulting erosion of trust harms consumers, penalises genuinely compliant farmers, and undermines the credibility of certification bodies.

Traditional supply chain record-keeping offers inadequate protection against such fraud. Centralised database solutions consolidate all records in one location, creating a single point of failure that an adversary—internal or external—can exploit [5]. Paper-based approaches are even more fragile, being susceptible to physical loss, alteration, and plain negligence. Neither model provides an independently verifiable audit trail that external parties can inspect without relying on the good-will of the record custodian.

Distributed ledger technology, and Ethereum in particular, addresses these weaknesses at an architectural level. Records written to the blockchain are replicated across hundreds of network nodes; altering any one record would require simultaneously overpowering the majority of those nodes—a practically infeasible attack under normal network conditions [1]. Smart contracts—self-executing programs stored on the chain—enforce business rules without requiring a trusted intermediary, and they execute identically on every node, making their output auditable by anyone with network access.

This paper presents a complete, working implementation of an organic supply chain traceability system built on these foundations. The six-role architecture (farmer, distributor, factory/processing unit, retailer, administrator, and consumer) mirrors the real stakeholder landscape of agricultural commerce. A hybrid storage model keeps gas costs

manageable by storing only essential immutable fields on-chain while retaining richer structured data in MongoDB. Consumer verification is deliberately frictionless: a smartphone camera and a QR code suffice to retrieve the complete product history.

The remainder of the paper is organised as follows. Section II surveys related work. Section III describes the system architecture. Section IV details the system design. Section V presents implementation specifics. Section VI compares the proposed approach with conventional methods. Section VII reports testing outcomes. Section VIII discusses achievements and limitations. Section IX concludes the paper.

II. LITERATURE SURVEY

Research into food supply chain traceability predates blockchain by several decades. Early computerised systems replaced handwritten logs with relational databases, which improved retrieval speed but did not eliminate the trust problem inherent in any centralised repository [5]. The introduction of radio-frequency identification (RFID) and Internet of Things (IoT) sensors in the early 2000s allowed real-time, automated data capture from farm equipment, refrigerated trucks, and warehouse shelves. This hardware innovation improved data granularity considerably, yet the collected data was still funnelled into centralised silos, leaving the integrity guarantee essentially unchanged [2].

The first prominent proposal to combine RFID with a distributed ledger for food safety traceability appeared in 2016 [2]. That work demonstrated measurable reductions in the time required to trace a contaminated batch back to its origin, and it attracted significant follow-on research. Tian subsequently refined the model to incorporate HACCP (Hazard Analysis and Critical Control Points) principles, aligning the blockchain record structure with existing food-safety regulatory frameworks [3]. Bhatt et al. [4] extended provenance tracking specifically to agricultural produce, providing empirical evidence that immutable ledger entries can effectively neutralise counterfeit-label fraud in commodity markets.

The scalability of public blockchain networks emerged as a recognised limitation during this period. On-chain transaction throughput is constrained by block size and block interval parameters, and each transaction incurs a gas fee that scales with computational complexity. Crosby et al. [5] documented these constraints and anticipated that hybrid architectures—where only critical immutable fields are stored on-chain while detailed records reside off-chain—would become the dominant practical pattern. Salah et al. [6] validated this prediction with a concrete implementation for soybean traceability, achieving a workable balance between immutability and query performance.

From an organic agriculture perspective, blockchain-based transparency has been linked to broader sustainability outcomes [7]. Verifiable evidence of sustainable practices reduces the information asymmetry between producer and consumer, which in turn can reduce overproduction and food waste driven by consumer scepticism. ISO 22005:2007 [8] provides the normative framework within which any food traceability system must operate, and blockchain architectures have been shown to satisfy its core requirements for record uniqueness, chronological ordering, and data accessibility.

The present work builds on this body of research by delivering a full-stack, deployable implementation that combines Ethereum smart contracts, a React.js consumer interface, MongoDB hybrid storage, and QR-based verification into a single coherent platform—addressing gaps identified in prior literature regarding end-to-end deployability and low-friction consumer access.

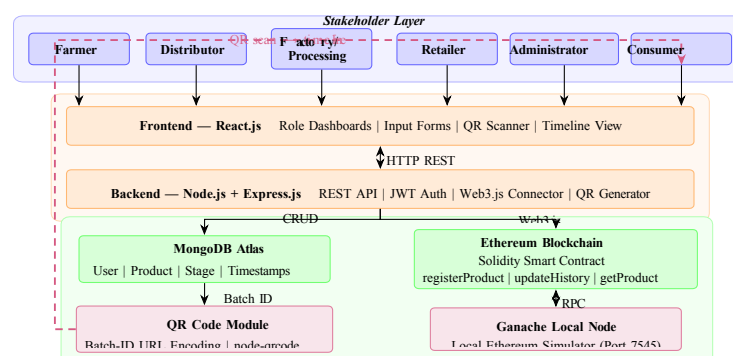


Fig. 1: System Architecture of the Organic Supply Chain Traceability Platform. *Blue region*: Stakeholder Layer (all six user roles); *Orange region*: Application Tier (React.js front- end and Node.js/Express.js back-end); *Green region*: Data & Persistence Layer (MongoDB Atlas, Ethereum blockchain, Ganache simulator). The dashed purple arrow represents the consumer QR-scan verification path.

III. SYSTEM ARCHITECTURE

A. Overview

The system follows a multi-tier client-server architecture in which a conventional web stack is tightly coupled with an Ethereum blockchain network. Six stakeholder categories interact with the platform: Farmer, Distributor, Factory/Processing Unit, Retailer, Administrator, and Consumer. Figure 1 illustrates the complete architecture across three distinct layers—the stakeholder layer at the top, the application tier in the middle, and the data persistence layer at the bottom.

B. Technology Stack

Table I lists the technologies employed at each architectural layer, together with the rationale for each selection.

TABLE I: Technology Stack

Component	Technology	Rationale
Blockchain platform	Ethereum (Ganache)	Mature smart-contract support
Smart contract language	Solidity	Native Ethereum language
Blockchain library	Web3.js	JavaScript-native JSON-RPC
Front-end	React.js	Component-based, reactive UI
Back-end	Node.js Express.js	+Non-blocking I/O for APIs
Database	MongoDB (Atlas)	Flexible document store
QR generation	node-qrcode	Server-side PNG/SVG output
Development tools	VS Code, Postman, Industry-standard toolchain Remix IDE	

C. Hybrid Storage Model

On-chain storage on Ethereum is permanent but costly in terms of gas fees and considerably slower than conventional database operations. The system therefore adopts a hybrid model: the Ethereum smart contract stores the minimum set of immutable fields required for fraud-resistance—Batch ID, product name, origin location, and the concatenated history string—while MongoDB retains the full structured record, including all stage sub-documents, actor identities, timestamps, and user credentials. This separation ensures that the anti-tamper guarantee of the blockchain is preserved for the most critical provenance data, while rich relational-style queries remain fast and cost-free.

SYSTEM DESIGN

D. User Roles and Access Control

The platform enforces role-based access control (RBAC) with six distinct roles, each mapped to a specific set of permitted operations. The supply chain lifecycle is started by the farmer role. A new product can be registered with a globally unique batch. ID product name farm name and place of origin. This. Transportation records are appended by the distributor including vehicle details. Identification of the logistics partner and the current transit coordinates—once. From the farm gate the product departs. The processing/factory. Processing events like cleaning and grading are recorded by the unit role. packaging as well as the outcomes of each transformational quality inspection. stage of transition. The retailer modifies the store and retail availability. location and is the only position with the authority to activate the QR code. creation for consumer-facing confirmation. Customers. role offers read-only access to the entire timeline of a product. by scanning a QR code no prior registration or login is required. needed. Lastly the

Administrator keeps an eye on everyone who has signed up. users and goods via a specific dashboard but are unable to alter any blockchain record while maintaining its integrity. the ledgers guarantee.

E. Smart Contract Design

The Ethereum smart contract is authored in Solidity and serves as the immutability backbone of the entire platform. internally. It keeps track of a mapping between a product and every Batch ID. structure that contains the product name place of origin and a. running history string as well as an existence flag that is boolean. Four. Every interaction on the chain is governed by public functions. The Batch ID is accepted by registerProduct the first function. product name place of origin and a preliminary data string sup-. supplied by the farmer registering. Prior to writing the agreement. confirms that the specified Batch ID is not already present in any records. thereby avoiding any unintentional or intentional overwriting. a well-known product launch. The data is released once the guard has passed. is permanently recorded on the blockchain. UpdateProductHistory the second function permits autho. stakeholders to be added at later stages of the supply chain. new details about the past of a product. Every call compiles. the new entry via a pipe to the current history string. delimiter creating a tamper-proof chronological order. chain of custody that expands but never changes with each stage. loses earlier data that was recorded.

getProduct the third function is a read-only view function that yields a result. the full history origin and name of a specific Batch ID. since you can see. Functions dont change the state of the blockchain and they dont cost any gas. Every time a customer timeline request is fulfilled the back-end is called freely.

F. Database Structure

MongoDB keeps track of four collections. The collection stores for User. the assigned role display username password hashed by bcrypt and user identification. each stakeholders registered name. It is stored in the Product collection. product name origin product identifier and Batch ID (imposed as a unique index). string as well as the timestamp of registration. Each is recorded in the Stage collection. individual custody transfer using the Batch ID to link it to its parent product. as well as recording the actors name role location and any relevant details. exact time stamp. The BlockchainRecord collection is kept up to date locally. mirror of the owner batch ID and on-chain history string. reference lowering how frequently the Ethereum node receives RPC calls. high-volume reading activities.

IV. IMPLEMENTATION

G. User Management Module

The user management module is responsible for account registration, authentication, and session control across all stakeholder roles. when something is fresh. Through React the user enters their login information. The registration form for JS. The back-end verifies that the selected username is unique in the MongoDB. user acquisition. The given password is hashed using the name if it is available. the bcrypt algorithm before being persisted—ensuring that raw passwords are. never sent or kept in a readable format. The server once the login is successful. gives the authenticated user a JSON Web Token (JWT) that contains the role they have been assigned. This token is carried by the front-end in all subsequent API requests. protected Express authorization header. Every request is intercepted by the js middleware. decodes the token and verifies that the embedded role and the permission match. requirement of the intended path. The request is denied if there is any discrepancy. before the business logic is reached right away. This design makes sure that a. A consumer and a farmers registration endpoint cannot be invoked by a distributor. There are no write operations that the account can access.

H. Registration of Farmer Product

The first step in establishing an on-chain presence is product registration. natural product. The registration form is filled out and submitted by a farmer. The farmers role is combined in an initial history entry created by the back-end. identifier display name and the batches descriptive data. The deployed Solidity smart contract receives this entry after that. Web-3. The Ethereum JSON-RPC is generated from the function call by the js library. The local Ganache node is the target of the transaction. The system intentionally writes to. the blockchain prior to sending data to MongoDB: if the on-chain transaction is unsuccessful. for any reason including inadequate gas or a duplicate Batch ID that was denied. contract guard—the two data stores are unable to do so since no MongoDB document is generated. descend into a state of instability. After the blockchain transaction that is. verified that the enriched product document is preserved in the back-end. first stage sub- document to the collection in MongoDB.

I. Module for Stage Update

Every subsequent custody transfer—whether by a distributor a factory or a. retailer—travels via a single stage-update

endpoint. starting with the back end. locates the product document in MongoDB using the supplied Batch ID if no. When a matching document is discovered a descriptive error is returned right away. When the. There is a new stage sub- document with the actors name and role. current location a timestamp produced by the server and descriptive information. appended to the products stage array and saved to MongoDB. Additionally concurrently. The back-end combines the ISO- 8601 timestamp to create a formatted history entry. the location the event description the uppercase role label—and submits. it to the updateProductHistory feature of the smart contract. the chain itself. The string then expands by a single pipe-delimited segment permanently documenting that phase. in a format that is not amendable or erasable. This has two writes. pattern ensures that every stage update is simultaneously reflected in both the. both the immutable blockchain ledger and a queryable MongoDB record.

J. Creating QR Codes

once a retailer has recorded the final supply and added a product to store stock. chain stage a verification is encoded in a scannable QR code created by the system. URL with the Batch ID of the product as a path parameter. There is an encoded URL. Using the server-side node-qrcode library a QR image can be created and returned. PNG data URL encoded in Base64. The retailer is presented with this image. dashboard and is meant to be printed on or attached to the tangible item. packaging. Sensitive information is purposefully avoided in the design. directly within the QR code the URLs Batch ID serves as a stand-in key. At scan time the back-end retrieves real-time data guaranteeing that the data is accurate. The most recent on-chain and MongoDB state is always displayed to the consumer.

K. Verification by customers

When a customer scans the QR code the camera app on their device fixes the issue. opens the verification page in a browser using the encoded URL—no application download is required. or creating an account is necessary. The response. A GET request is sent to the front-end of js. timeline endpoint passing the URL-extracted Batch ID. back-end. sends two requests at once one to the Ethereum smart contract. `emphgetProduct` view function to obtain the on-chain record (name) that cannot be changed. origin and history string) and one to retrieve the complete structured data from MongoDB. array stage. The two answers are combined into one JSON object and sent back. the front-end which presents a chronological representation of the entire product journey. timeline. The tamper-evident anchor is represented by the on-chain history string. Actor names locations and other richer details are provided by MongoDB stage records. timestamps—which enable a non- technical person to read and understand the timeline. audience.

L. Blockchain Setup

The Node when the server is starting up. The js back-end connects to Ganache. local Ethereum simulator by creating a Web3 provider with the default address. 7545 is the Ganache RPC endpoint. The Application Binary Interface (ABI) after compilation. is read from a local JSON file created during the deployment of the smart contract. Compile the Remix IDE step. Web3. JS makes use of both the deployed and the ABI. contract address—which is kept as an environment variable—to generate a JavaScript. contract object that makes every Solidity function available as an awaitable method. All of them. This is how the applications subsequent blockchain interactions are routed. single contract object maintaining a clean blockchain communication layer. detached from the Express business logic. handlers for js routes.

V. COMPARISON: EXISTING VS. PROPOSED SYSTEM

TABLE II: Existing System vs. Proposed System

Parameter	Existing System	Proposed System
Data integrity	Modifiable or deletable	Immutable, tamper-proof
Trust model	Intermediary-dependent	Blockchain-enforced, trustless
Traceability	Partial or fragmented	End-to-end, farm to consumer
Security	Single point of failure	Cryptographic, distributed
Consumer verification	Manual or unavailable	QR code, real-time, no login
Fraud detection	Difficult; requires manual audit	Anomalies visible on-chain
Automation	Mostly manual data entry	Smart contract-enforced updates
Scalability	Limited by central server	Modular, decentralised architecture

TABLE III: Functional Test Cases and Results

ID	Scenario	Expected Output	Pass
TC01	User registration with valid credentials	withSuccess message	✓
TC02	Login with credentials	validRole dashboard shown	✓
TC03	Login with incorrect pass- word	incorrectError message	✓
TC04	Product registration with unique Batch ID	Stored in DB & chain	✓
TC05	Duplicate Batch submission	IDRegistration rejected	✓
TC06	Stage update with valid Batch ID	withStage appended	✓
TC07	Full product query	trackingJourney displayed	✓
TC08	Query with invalid Batch ID	Error message shown	✓
TC09	QR code scan (QR code)	(validProduct details shown	✓
TC10	Administrator dashboard access	User list displayed	✓

VI. TESTING AND RESULTS

M. Functional Testing

Ten test cases were executed to verify the correctness of every major system workflow. Table III summarises the scenarios, inputs, expected outputs, and outcomes.

All ten test cases passed without critical errors, confirming that the implemented workflows align with the specified functional requirements.

N. Non-Functional Testing

Non-functional properties were evaluated across four dimensions.

Performance. Back-end API responses for MongoDB-only queries—such as user login and stage listing—completed within acceptable latency bounds. Blockchain write operations exhibited higher latency due to the transaction confirmation cycle on Ganache, which was expected and mitigated by the hybrid storage model: consumers and administrators interact primarily with MongoDB for data retrieval, invoking the blockchain only for immutability verification.

Security. Role-based access control was verified by attempting cross-role operations, for example a distributor account attempting to register a product. All such attempts were rejected at the API layer. On-chain records were confirmed to be unmodifiable after registration by attempting direct contract calls with altered data—the duplicate Batch ID guard prevented any overwriting.

Usability. The QR-based verification flow was tested on both Android and iOS devices using default camera applications. Product timelines rendered correctly in both Google Chrome and Microsoft Edge across desktop and mobile form factors.

Scalability. The decoupled architecture—where the React front-end, Node.js back-end, MongoDB Atlas, and Ganache Can each be scaled or replaced independently—supports horizontal scaling of the application tier and seamless migration to a cloud-hosted or permissioned blockchain without architectural redesign.

VII. DISCUSSION

O. Key Achievements

The implemented system demonstrates that Ethereum-based traceability is practically achievable within the constraints of an academic development environment. Three outcomes are particularly noteworthy.

First, the hybrid storage model proved effective in practice. Gas costs remained predictable because the on-chain payload was restricted to a minimal set of fields; MongoDB absorbed all querying load, keeping response times acceptable.

Second, the QR-based consumer interface removes the principal barrier that has historically limited blockchain-based traceability to supply chain professionals. Any consumer with a smartphone can verify a product's organic credentials within seconds, with no account creation or technical knowledge required.

Third, the role-based architecture scales naturally to the real stakeholder map of agricultural commerce. Adding a new role—such as a certifying body or a customs inspector—requires only a new role definition in the RBAC middleware and a corresponding dashboard view in the React front-end; the underlying smart contract and database schema remain unchanged.

P. Limitations

The current prototype operates against Ganache, a local Ethereum simulator that does not enforce real gas fees and does not expose the system to network adversaries. Deploying to a permissioned network like the Ethereum mainnet. Transaction costs and throughput limitations would be introduced by Hyperledger Fabric, that need cautious handling. Additionally all data entry currently depends on people involved in every stage of the supply chain. A malicious or careless actor might input inaccurate data which the blockchain would then keep information indefinitely. Therefore integrating IoT sensors is a crucial next step.

Q. Future Upgrades

For later development cycles a number of extensions are planned:

- **IoT Sensor Integration:** Temperature is automatically recorded. GPS coordinates weight information and humidity at every custody change, removing the need for human input.
- **Mobile Applications:** Farmers and distributors specific apps, and retailers to send in stage updates straight from the field.
- **AI/ML Analytics:** Anomaly detection models trained on historical supply chain data to identify potentially problematic stage updates prior to their commit to the chain.
- **Certification Body Integration:** Digital connection to government, and independent organic certification authorities in order to obtain certification status. The product record automatically reflects this.
- **Multi-chain Support:** Hyperledger Fabric Investigation. Polygon as a substitute ledger platform for deployment on a large scale.
- **Multi-language Support:** The consumer interface is localized, for Indian rural and regional stakeholders as well as other farmers, the economy.

VIII. IN SUMMARY.

The entire design implementation and testing of an has been described in this paper. An Ethereum blockchain-based system for tracking organic supply chains, trustworthy core. Conventional supply chains were the main issue that was addressed, cannot guarantee as organic status in a way that can be independently verified, product—is a result of the traditional records centralized and changeable nature, mechanisms, by creating an unchangeable blockchain that records each custody transfer, transaction and making that record available to customers via a frictionless QR code, interface the institutional trust model is replaced by the suggested system, reliance on cryptographic verifiability. Each participant in a realistic organic is accommodated by the six-role architecture, supply network. The Ethereum-MongoDB hybrid storage model strikes a balance between permanence and, the performance. The system was verified by the passing of all ten functional test cases, fulfills its requirements, limitations pertaining to manual data and gas prices, entry point to well-defined routes for future IoT integration development, transition to blockchain networks with permissions. When taken as a whole these findings show that blockchain technology is advanced, sufficient to handle practical traceability issues in organic agriculture, and that useful implementation—assisting rural Indian farmers as easily as possible, consumers in urban centres—is achievable with the technology stack described in, this essay.

REFERENCES

- [1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. [Online]. Available: <https://bitcoin.org/bitcoin.pdf>
- [2] F. Tian, "An Agri-food Supply Chain Traceability System for China Based on RFID & Blockchain Technology," in *Proc. 13th Int. Conf. Service Systems and Service Management (ICSSSM)*, 2016, pp. 1–6.
- [3] F. Tian, "A Supply Chain Traceability System for Food Safety Based on HACCP, Blockchain & Internet of Things," in *Proc. Int. Conf. Service Systems and Service Management*, 2017, pp. 1–6.
- [4] S. Bhatt, R. Bhatt, and D. Mehta, "A Blockchain-Based Framework for Agri-Product Traceability," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 6, pp. 342–348, 2020.
- [5] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain Technology: Beyond Bitcoin," *Appl. Innov.*, vol. 2, pp. 6–10, 2016.
- [6] K. Salah, N. Nizamuddin, R. Jayaraman, and M. Omar, "Blockchain- Based Soybean Traceability in Agricultural Supply Chain," *IEEE Access*, vol. 7, pp. 73295–73305, 2019.
- [7] World Economic Forum, "Blockchain for Supply Chains: A Practical Guide," 2020. [Online]. Available: <https://www.weforum.org>
- [8] International Organization for Standardization, "Food Traceability in the Supply Chain—General Principles and Basic Requirements," ISO 22005:2007, 2007.
- [9] Ethereum Foundation, "Ethereum Developer Documentation." [Online]. Available: <https://ethereum.org/developers/>
- [10] MongoDB Inc., "MongoDB Documentation." [Online]. Available: <https://www.mongodb.com/docs/>

International Journal of Advanced Research in Education and Technology

ISSN: 2394-2975

Impact Factor: 8.152