



Journal of Interdisciplinary Science & Technology

Blockchain-based e-commerce warranty system using non-fungible tokens (NFTs)

A Smart Contract Framework for Fraud-Resistant and Transferable Product Warranties

Naresh Thoutam¹, Yogesh Baviskar², Ritu Garge², Dipesh Gavit², Ravina Bhoje²

¹Assistant Professor, Department of Computer Engineering, Sandip Institute of Technology and Research Center, Nashik, India

²Department of Computer Engineering, Sandip Institute of Technology and Research Center, Nashik, India

naresh.thoutham@sitrc.org, yogeshbaviskar0603@gmail.com, gargeritu17@gmail.com, dipeshgavit333@gmail.com,
ravinabhoye0007@gmail.com

Corresponding author- Naresh Thoutam

DOI: 10.5281/zenodo.23438282

Abstract: Traditional e-commerce warranty systems rely on paper receipts or centrally stored digital records that are easily lost, forged, or disputed, and that do not survive a change of ownership when a product is resold. This paper proposes a blockchain-based warranty management system that represents each product warranty as a Non-Fungible Token (NFT) minted to the buyer's wallet at the time of purchase. Built on the ERC-721 token standard and deployed on a low-cost, high-throughput blockchain such as Polygon, the proposed system binds warranty terms, purchase metadata, and claim history to an immutable, publicly verifiable token whose ownership transfers alongside the physical product during resale. A smart contract enforces role-based minting, claim filing and approval, automatic expiry, and revocation, while bulky metadata such as invoices and images is stored off-chain on IPFS and referenced on-chain through a content hash. This paper presents the system architecture, smart contract interface, implementation approach, security analysis, and an evaluation methodology, together with a comparative analysis against conventional warranty systems on fraud-resistance, verification effort, and transferability. The proposed design indicates that an NFT-based warranty layer can substantially reduce warranty fraud and manual verification overhead while introducing a transferable, tamper-evident record that benefits both consumers and manufacturers, particularly in the growing second-hand electronics market.

Keywords: Blockchain, Non-Fungible Token (NFT), Smart Contract, E-commerce, Warranty Management, ERC-721, Ethereum, Polygon, IPFS.

I. INTRODUCTION

The global e-commerce industry has grown into a multi-trillion-dollar market, and with it the volume of products sold with an accompanying manufacturer or seller warranty has grown proportionally. A product warranty is, in effect, a trust contract between buyer and seller: it promises repair, replacement, or refund under defined conditions for a defined period. In practice, however, warranties are still issued as paper cards, printed receipts, or, at best, unstructured digital records (PDFs or database entries controlled by a single company). These artefacts are easy to lose, easy to forge, and difficult to verify independently. They also do not travel with the product: when a buyer resells a laptop, a phone, or an appliance on a second-hand marketplace, the warranty almost always stays tied to the original purchaser's account or is

simply lost, even though the manufacturer's coverage period has not expired.

Blockchain technology offers three properties that map directly onto these problems: immutability (a warranty record cannot be silently altered once issued), transparency (any party can independently verify a claim without trusting a single intermediary), and disintermediated transfer (ownership of a digital asset can move between wallets without the involvement of the issuing company's database). Within blockchain systems, Non-Fungible Tokens (NFTs) are the natural data structure for representing a warranty, because every warranty instance is unique: it is tied to a specific serial number, purchaser, purchase date, and expiry, unlike a fungible token such as a cryptocurrency where every unit is interchangeable with every other unit.

This paper proposes and describes the design of a **blockchain-based e-commerce warranty system** in which each purchased product is issued an ERC-721 NFT that acts as its digital warranty card. The NFT is minted to the buyer's wallet at checkout, can be presented to file a claim, can be transferred to a new owner on resale so that the remaining warranty period carries over automatically, and expires or can be revoked according to on-chain logic. The remainder of this paper is organised as follows: Section II reviews related work; Section III states the problem and objectives; Section IV presents the proposed system architecture; Section V describes the implementation approach; Section VI analyses the security and fraud-resistance properties of the design; Section VII compares the proposed system with conventional warranty mechanisms; Section VIII outlines the evaluation methodology; and Section IX concludes the paper with directions for future work.

II. RELATED WORK

The foundational concepts used in this work originate from Nakamoto's proposal of a peer-to-peer electronic cash system secured by a distributed, immutable ledger [1], and from Buterin's extension of this idea into a general-purpose, Turing-complete smart contract platform, Ethereum [2]. The specific token interface used to represent unique, non-interchangeable digital assets on Ethereum-compatible chains is standardised as ERC-721 [3]. Chirtoaca et al. proposed a reusable, configurable smart contract suite implementing ERC-721 and its common extensions, which informs several design choices made for the warranty contract in this paper, including the separation of core minting logic from optional extensions such as enumeration and metadata [4].

Closest to the present work, Ali et al. implemented an NFT-based warranty system as a software-as-a-service offering and integrated it into a cloned e-commerce website to study ease of use; their system supports automatic warranty issuing, verification, ownership transfer, and expiration, and demonstrates that warranties from multiple stores can be aggregated into a single wallet-linked view for the consumer [5]. Meel and Jangid examined NFT-based warranties deployed on the NEAR blockchain and compared transaction throughput, scalability, and gas cost against alternative blockchains, highlighting that chain selection materially affects the practicality of a warranty-per-purchase minting model [6]. Outside the warranty domain specifically, a broader body of work has used blockchain to combat counterfeiting and establish product provenance in supply chains: Alzahrani and Bulusu combined blockchain with Near Field Communication (NFC) tags in a decentralised anti-counterfeiting supply chain [7]; Ma et al. proposed a blockchain-based application for product anti-counterfeiting using cryptographic product identifiers [8]; and Jayaprasanna

et al. designed a blockchain-backed management system that links a barcode reader to an on-chain ledger to detect counterfeit goods [9].

Taken together, this prior work establishes that (a) NFTs are a technically sound representation for unique, ownership-bearing warranty records, and (b) blockchain-based product-authentication systems are well studied in the anti-counterfeiting literature. However, comparatively few of the reviewed systems combine a documented, role-based claim-approval workflow, resale-transfer mechanics with automatic warranty carry-over, and a quantitative gas-cost and security evaluation within a single, deployable architecture. This paper's contribution is to bring these elements together into one coherent system design and evaluation plan.

III. PROBLEM STATEMENT AND OBJECTIVES

Existing e-commerce warranty mechanisms suffer from four recurring problems: (i) warranty proof is easily lost or damaged when issued on paper; (ii) digital warranty records held in a single company's database are opaque to the buyer and can be altered without an audit trail; (iii) warranty coverage does not transfer automatically or verifiably when a product changes hands in the resale market; and (iv) fraudulent claims are difficult to detect because there is no independently verifiable link between a specific physical unit and its warranty history. Based on these problems, this project defines the following objectives:

1. Design an ERC-721-based smart contract that mints one NFT per purchased product, binding warranty terms, product serial number, and purchase date to an immutable on-chain record.
2. Support a documented, role-restricted workflow for filing, approving, and rejecting warranty claims, with every state change recorded on-chain.
3. Allow warranty ownership to be transferred between wallets on resale, so that remaining coverage follows the product rather than the original buyer.
4. Keep on-chain storage and transaction costs low by storing bulky metadata off-chain on IPFS and referencing it via a content hash.

IV. PROPOSED SYSTEM ARCHITECTURE

The proposed system involves three primary roles: the **Manufacturer/Seller**, who is authorised to mint warranty NFTs and approve or reject claims; the **Buyer**, who receives, holds, and can transfer the warranty NFT and who files claims against it; and an optional **Administrator**, who can intervene in disputes or void a warranty in cases of confirmed fraud. The system is organised into three logical layers, shown in Fig. 1.

The **blockchain layer** consists of a single ERC-721 smart contract deployed on a Polygon test network, chosen over

Ethereum mainnet for its substantially lower transaction fees, which is important in a model that mints one token per purchase. The **storage layer** uses IPFS (pinned via a service such as Pinata) to hold warranty metadata that would be expensive to store on-chain, including the invoice image, extended terms and conditions, and product photographs; only the resulting content hash is stored inside the NFT's metadata URI field on-chain. The **application layer** consists of a backend service that listens for contract events and mirrors warranty state into a conventional database for fast querying, together with a buyer-facing and seller-facing web frontend connected to a browser wallet such as MetaMask.

The end-to-end workflow proceeds as follows: (1) at checkout, the seller's backend calls the contract's minting function, which creates a new token bound to the buyer's wallet address, the product's serial number, and the warranty expiry timestamp; (2) the token remains in the buyer's wallet as proof of ownership and coverage, visible through the frontend regardless of which participating store issued it; (3) to file a claim, the buyer submits the token ID together with a description of the fault, which the seller or administrator reviews and approves or rejects on-chain; (4) if the buyer resells the product, the NFT is transferred to the new owner's wallet using the contract's transfer function, and the remaining warranty period is preserved automatically because it is stored as an absolute expiry timestamp rather than a duration relative to the original owner; and (5) once the expiry timestamp has passed, or if fraud is confirmed, the token is marked invalid or void and can no longer be used to file a claim.

Table 1 summarises the core functions exposed by the warranty smart contract.

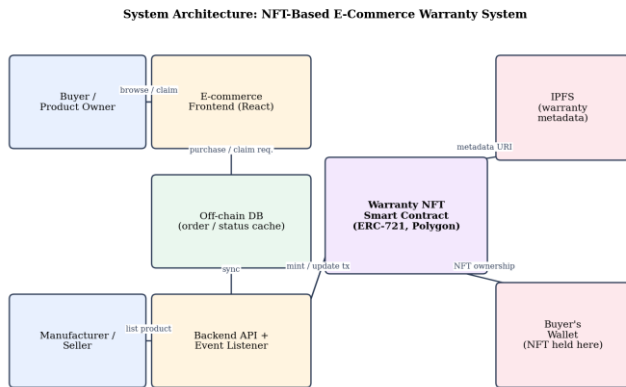


Figure 1: Layered architecture of the proposed NFT-based warranty system

Table 1: Core Smart Contract Functions

Function	Caller Role	Description
mintWarranty()	Manufacturer / Seller	Mints a new warranty NFT to the buyer's wallet with serial number, expiry timestamp, and metadata URI.
fileClaim()	Buyer (token owner)	Opens a claim against a token that is still within its warranty period.
approveClaim() / rejectClaim()	Manufacturer / Seller / Admin	Resolves a pending claim and records the outcome on-chain.
transferWarranty()	Buyer (token owner)	Transfers the NFT, and the remaining coverage it represents, to a new owner on resale.
isValid()	Any (view function)	Returns whether a given token is currently within its warranty period and not void.
voidWarranty()	Admin	Permanently invalidates a token in confirmed cases of fraud or tampering.

V. IMPLEMENTATION

The prototype is implemented using the technology stack summarised in Table 2. The smart contract extends

OpenZeppelin's audited ERC-721 and AccessControl base contracts rather than re-implementing the token standard from first principles, which reduces the attack surface of the custom code to the warranty-specific logic described in Section IV.

Table 2: Technology Stack

Layer	Tool / Technology
Smart contract	Solidity, OpenZeppelin Contracts (ERC-721, AccessControl)

Layer	Tool / Technology
Development & testing	Hardhat, Chai/Mocha, Hardhat Gas Reporter
Blockchain network	Polygon (Amoy test network for evaluation)
Off-chain storage	IPFS, pinned via Pinata
Backend	Node.js, Express, ethers.js (event listeners)
Off-chain cache	MongoDB / PostgreSQL
Frontend	React, ethers.js, MetaMask wallet connector
Product binding	QR code on physical product linking to on-chain verification page

Each warranty's off-chain metadata follows a fixed JSON schema pinned to IPFS: product name, category, brand, serial number, invoice reference, purchase date, warranty duration in months, and an array of claim history entries, each with a timestamp, description, and resolution status. Only the resulting IPFS content identifier (CID) is written on-chain as Role separation is enforced using OpenZeppelin's AccessControl module with three roles: **MANUFACTURER_ROLE**, permitted to call `mintWarranty()`; **SELLER_ROLE**, permitted to mint on behalf of a manufacturer under a delegated agreement and to resolve claims; and **ADMIN_ROLE**, permitted to void a token and to reassign roles. This mirrors the principle of least

VII REFERENCES:

- [1] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.
- [2] V. Buterin, "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform," Ethereum White Paper, 2014.
- [3] W. Entriiken, D. Shirley, J. Evans, and N. Sachs, "EIP-721: Non-Fungible Token Standard," Ethereum Improvement Proposals, no. 721, Jan. 2018.
- [4] D. Chirtoaca, J. Ellul, and G. Azzopardi, "A framework for creating deployable smart contracts for non-fungible tokens on the Ethereum blockchain," in Proc. IEEE Int. Conf. Decentralized Applications and Infrastructures (DAPPS), Oxford, U.K., Aug. 2020, pp. 100–105.
- [5] A. Ali, S. Agrawal, and S. Dongre, "Blockchain-Based NFT Warranty System: A Software Implementation," Journal of Comprehensive Business Administration Research, vol. 1, no. 1, pp. 12–18, 2024.
- [6] P. Meel and H. Jangid, "NFT Based Warranty Using NEAR Blockchain," SSRN Electronic Journal, May 2023.
- [7] N. Alzahrani and N. Bulusu, "Block-Supply Chain: A New Anti-Counterfeiting Supply Chain Using NFC and Blockchain," in Proc. 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems, ACM, 2018.

the token's metadata URI, which keeps per-mint gas cost close to the minimum required for an ERC-721 mint. A QR code printed on or attached to the physical product encodes the token ID and contract address, allowing any party, including a prospective second-hand buyer, to look up the token's current owner and validity without needing a crypto wallet.

privilege: a compromised seller account cannot, for example, void warranties issued by a different seller.

VI ACKNOWLEDGMENT

The authors thank Prof. Naresh Thoutam, Department of Computer Engineering, SITRC, Nashik, for their guidance and support throughout this project.

- [8] J. Ma, S.-Y. Lin, X. Chen, H.-M. Sun, and Y.-C. Chen, "A Blockchain-Based Application System for Product Anti-Counterfeiting," IEEE Access, vol. 8, pp. 77642–77652, 2020.
- [9] M. C. Jayaprasanna, V. A. Soundharya, M. Suhana, and S. Sujatha, "A Block Chain based Management System for Detecting Counterfeit Product in Supply Chain," in Proc. 2021 3rd Int. Conf. Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), Feb. 2021.
- [10] OpenZeppelin, "OpenZeppelin Contracts Documentation," [Online]. Available: <https://docs.openzeppelin.com/contracts>. Accessed: 2026.
- [11] Polygon Labs, "Polygon PoS Documentation," [Online]. Available: <https://docs.polygon.technology>. Accessed: 2026