STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 202x, pp 0–22.
Published online in International Academic Press (www.IAPress.org)

Analyzing Smart Inventory Management System Performance Over Time with State-Based Markov Model and Reliability Approach, Enhanced by Blockchain Security and Transparency

Nabil Chbaik [1,*], Azeddine Khiat [1], Ayoub Bahnasse [2], Hassan Ouajji [1]

[1] *2IACS Laboratory, ENSET, Hassan II University of Casablanca, Mohammedia, Morocco*
[2] *ENSAM, Hassan II University of Casablanca, Casablanca, Morocco*

Abstract In response to the evolving demands of modern inventory management, this paper introduces a quantitative mathematical model aimed at assessing the performance of a smart inventory management system, emphasizing the integration of blockchain technology to enhance security and transparency. By considering essential hardware components, including ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires, the model utilizes Markov modeling and a reliability-based approach. Its primary objective is to enable timely repair and maintenance activities, ensuring sustained system availability by identifying and addressing weak components. Efficient operation of all system parts is critical for the timely transmission of inventory data. Through the incorporation of blockchain technology, the study addresses security and transparency concerns, alongside evaluating key reliability metrics such as reliability, unreliability, and Mean Time to Failure (MTTF). Sensitivity analysis identifies critical components, highlighting the importance of maintenance for components like the switch and servo motor. The research underscores the role of blockchain technology in fortifying security and transparency in smart inventory management systems, alongside emphasizing the significance of reliability metrics in system performance optimization.

Keywords Smart Supply Chain Management, Smart Inventory System, Internet of Things, Blockchain, Arduino

DOI: 10.19139/soic-2310-5070-2127

1. Introduction

In recent years, the landscape of inventory management has witnessed a significant transformation, largely driven by advancements in technology and the increasing demand for efficiency, accuracy, and security. Traditional inventory management systems have gradually given way to smart inventory management systems, which leverage technologies such as Internet of Things (IoT), automation, and data analytics to streamline operations and enhance decision-making processes. However, while these systems offer numerous benefits, they also pose new challenges, particularly in terms of reliability, security, and transparency. One of the critical challenges facing smart inventory management systems is the reliability of their hardware components. Unlike traditional systems, smart inventory management systems rely heavily on various hardware components, such as sensors, actuators, and communication devices, to collect, process, and transmit data in real-time. Any failure or malfunction of these components can lead to disruptions in operations, resulting in inventory inaccuracies, delays, and ultimately, financial losses [12]. Therefore, ensuring the reliability and availability of these components is paramount to the success of smart inventory management systems. To address this challenge, this paper proposes a novel quantitative mathematical

*Correspondence to: Nabil Chbaik (Email: nabil.chbaik7@gmail.com). 2IACS Laboratory, ENSET, Hassan II University of Casablanca, Mohammedia, Morocco.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

model that evaluates the performance of smart inventory management systems from a reliability perspective. The model considers a range of essential hardware components commonly used in such systems, including the ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires. By employing Markov modeling and a reliability-based approach, the model aims to identify critical components prone to failure and prioritize maintenance activities accordingly. Central to our approach is the integration of blockchain technology, which serves to enhance the security and transparency of smart inventory management systems. Blockchain, originally developed as the underlying technology for cryptocurrencies, has gained significant attention across various industries for its ability to provide a secure and tamper-proof ledger of transactions. In the context of inventory management, blockchain offers several advantages, including immutability, decentralization, and transparency. By recording all inventory-related transactions on a distributed ledger, blockchain ensures data integrity and fosters trust among stakeholders, mitigating the risk of fraud or unauthorized access. Furthermore, this research evaluates key reliability metrics such as reliability, unreliability, and Mean Time to Failure (MTTF) to quantify the performance of smart inventory management systems [5]. Through sensitivity analysis, we identify critical components that significantly impact system reliability and highlight the importance of proactive maintenance strategies. For instance, our findings indicate that components such as switches and servo motors are particularly sensitive to failure and require regular monitoring and maintenance to prevent disruptions in operations. In summary, this paper contributes to the existing body of literature on smart inventory management systems by introducing a comprehensive mathematical model that integrates blockchain technology to enhance security and transparency. By emphasizing reliability metrics and proactive maintenance strategies, our research aims to optimize the performance of smart inventory management systems, ensuring their reliability and availability in the era of Industry 4.0.

1.1. Proposed system

In this paper, the authors have considered the essential hardware components commonly found in smart inventory management systems: ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires, for the evaluation of the performance of the system. In this context, the authors have employed constant failure and repair rates for these components to formulate the system's model. All these components have only two possible states, "working" and "failed". Similarly, the system itself has only two states, "working" and "failed". Upon the failure of any component, the system goes into a failed state, and a proper repair facility is offered to bring the system back into a good working state. Therefore, throughout the system's entire lifecycle, it continuously alternates between these two states. This approach provides a comprehensive understanding of the system's behavior and enables the prediction of system reliability and performance over time. The hardware components of the smart inventory management system serve the following functions:

- ESP32 module: Acts as the central processing unit, managing data flow, processing sensor inputs, and controlling system operations. It coordinates various tasks such as receiving data from sensors, analyzing data, and sending commands to actuators.
- HC-SR04 ultrasonic sensor: Detects the presence and movement of items within the inventory storage area. It measures the distance between the sensor and the objects, providing accurate tracking and enabling automated inventory management.
- Battery: Provides power supply for continuous operation of the system. It ensures that the system remains operational even in the absence of external power sources, allowing for uninterrupted monitoring and control.
- Jumper wires: Facilitate the flow of data and electricity between different components of the system. They establish connections between the ESP32 module, sensors, actuators, and power source, enabling seamless communication and functionality.

Functionally, the system acquires and processes data in real-time, controls inventory access, monitors component health for predictive maintenance, and integrates blockchain for secure and transparent record-keeping. Operations include continuous monitoring of inventory, access control based on credentials, maintenance alerts, and blockchain-recorded transactions, ensuring data integrity and system efficiency.

1.2. Problem statement

In traditional inventory management systems, there are often challenges related to accuracy, efficiency, security, and transparency. These challenges can lead to inventory discrepancies, delays in replenishment, unauthorized access, and difficulties in maintaining reliable records of inventory transactions. Additionally, the reliance on manual processes for inventory control and management can be time-consuming and prone to human error. To address these issues, there is a need for a smart inventory management system that can provide real-time monitoring, automated inventory control, enhanced security, and transparent record-keeping. However, developing such a system requires addressing several key challenges:

- Reliability of Hardware Components: The hardware components used in smart inventory management systems, such as the ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires, must operate reliably to ensure continuous system functionality. Identifying critical components and implementing maintenance strategies to prevent failures is essential.
- Real-Time Data Acquisition and Processing: The system must be able to collect, process, and analyze inventory data in real-time to provide accurate and up-to-date information. The ESP32 module processes data from the HC-SR04 ultrasonic sensor to provide real-time inventory information.
- Access Control and Security: Ensuring that only authorized personnel have access to inventory items is crucial for preventing theft, unauthorized access, and tampering. Implementing robust access control mechanisms and maintaining data security are essential components of the system.
- Maintenance and Monitoring: Proactively monitoring the health and performance of system components is essential for preventing unexpected failures and minimizing downtime. Predictive maintenance strategies can help identify potential issues before they lead to system failures. The system continuously monitors the health of components and employs predictive maintenance techniques to ensure reliability.
- Data Integrity and Transparency: Maintaining accurate and transparent records of inventory transactions is essential for ensuring trust and accountability in the system. The system utilizes the ESP32 module to integrate blockchain for secure and transparent record-keeping, ensuring data integrity and accountability.

2. Literature review

Various studies have been conducted in the field of smart inventory management systems, focusing on different aspects such as hardware design, software algorithms, and system optimization. This section provides an overview of relevant literature, highlighting key contributions and areas of research as shown in Table 1.

The studies collectively represent a diverse range of approaches and applications in inventory management. Saha et al. and Ahakonye et al. leverage advanced computational techniques such as stochastic modeling and machine learning to optimize inventory strategies, focusing on healthcare and manufacturing, respectively. Bose et al. and Li propose integrating emerging technologies like IoT and blockchain into inventory management systems, aiming for broader applicability across sectors. Giovanni explores the implementation of smart applications to enhance Vendor Managed Inventory (VMI) in dynamic supply chains, while Affonso et al. develop a hybrid forecasting method to address irregular demand patterns in spare part inventory management, particularly in mining. Despite their differences, all studies aim to improve efficiency, reduce costs, and enhance decision-making in inventory management, underscoring the importance of technology integration and innovative approaches in this field.

3. Materials and methods

3.1. System description

The smart inventory management system comprises four essential hardware components: ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires. Initially, all components operate properly, and the failure of any single component results in the system's failure. Repair or replacement services are available to restore the system

Table 1. Contributions of recent studies in inventory management

Study	Key Contribution
Saha et al. (2024) [11]	This research introduces a stochastic semi-Markov decision process model to manage medicines in hospital supply chains (HSC), tackling challenges such as uncertain medication demand and dependencies among prescribed medicines. Using multi-agent reinforcement learning, the study identifies optimal inventory strategies for HSCs, improving inventory performance, healthcare service delivery, and cost reduction to ensure affordable and quality healthcare.
Bose et al. (2022) [3]	The abstract emphasizes the significance of efficient inventory management in manufacturing for sustainability and profitability. It suggests integrating barcodes with Cloud Computing, Arduino-based nodes, IoT, and secure web portals for better management. The proposed model targets inventory of formwork shuttering products in the construction sector, with potential applicability beyond Indian companies.
Ahakonye et al. (2024) [2]	Artificial intelligence (AI) is revolutionizing inventory management in industrial processes, particularly in manufacturing execution systems (MES). A Multi-MLP model with LightGBM feature selection is proposed for MES inventory prediction, aiming for high accuracy and efficiency. The model is tested using public datasets, demonstrating low prediction errors (MAE of 0.2331, MSE of 0.1225, and RMSE of 0.3504), highlighting its efficient decision-making capabilities.
Giovanni (2021) [4]	This paper examines the benefits of implementing smart applications and intelligent systems to enhance the efficiency of Vendor Managed Inventory (VMI) in a dynamic brick-and-mortar Supply Chain (SC). In this setup, the manufacturer determines production rates, while the retailer sets prices affecting sales and inventory. The study investigates the impact of smart applications on error reduction and consumer satisfaction, considering associated fees and environmental implications. It identifies conditions where smart system adoption is economically beneficial for the SC and discusses operational, environmental, and coordination implications.
Li (2023) [6]	This study presents a framework for blockchain-based supply chains, highlighting five key changes: a common platform, end-to-end information sharing, real-time updates, efficient asset transfer, and contract automation. It emphasizes how blockchain transforms supply chains into decentralized platforms, quantifies benefits in inventory management and information sharing, and discusses managerial implications for supply chain resilience, including efficiency gains and reduced risks for companies and partners. These findings are important for advancing smarter and more efficient supply chains as blockchain technology develops.
Affonso et al. (2024) [1]	This paper introduces a hybrid forecasting method for spare part inventory management, aiming to address the challenges of forecasting irregular demand patterns. By combining heuristics and bootstrapping approaches, it improves accuracy in spare parts forecasting, particularly during normal use phases. The study proposes an innovative methodology to model demand autocorrelation, enhancing data representation in bootstrapping. Results from a case study in a large Brazilian iron ore corporation demonstrate up to a 40% reduction in total costs compared to leading-edge techniques, especially for erratic and lumpy demand. The method provides a systematic tool applicable across sectors, particularly in mining, aiding in better spare parts management and cost reduction.

after a component failure. Notably, each component failure is independent, and only one failure can occur at a time. The smart inventory management system functions as follows: it provides real-time monitoring, automated inventory control, enhanced security, and transparent record-keeping. However, developing such a system requires addressing several key challenges:

- Reliability of Hardware Components: The hardware components used in smart inventory management systems, such as the ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires, must operate reliably to ensure continuous system functionality. Identifying critical components and implementing maintenance strategies to prevent failures is essential.
- Real-Time Data Acquisition and Processing: The system collects, processes, and analyzes inventory data in real-time to provide accurate and up-to-date information. The ESP32 module processes data from the HC-SR04 ultrasonic sensor to provide real-time inventory information.
- Access Control and Security: Ensuring that only authorized personnel have access to inventory items is crucial for preventing theft, unauthorized access, and tampering. Implementing robust access control mechanisms and maintaining data security are essential components of the system.
- Maintenance and Monitoring: Proactively monitoring the health and performance of system components is essential for preventing unexpected failures and minimizing downtime. Predictive maintenance strategies can help identify potential issues before they lead to system failures. The system continuously monitors the health of components and employs predictive maintenance techniques to ensure reliability.
- Data Integrity and Transparency: Maintaining accurate and transparent records of inventory transactions is essential for ensuring trust and accountability in the system. The system utilizes the ESP32 module to integrate blockchain for secure and transparent record-keeping, ensuring data integrity and accountability.

Assumptions for the Proposed Model:

- The system's modeling is based on several assumptions:
- Initially, all components are operational.
- System failure can occur due to any individual component failure.
- Repair and failure rates follow an exponential distribution.
- Component failures are independent.
- Proper repair facilities are available after a component failure.

Table 2 provides a description of the symbols and notations used throughout the paper.

Table 2. Notations and system's various states

Notation	Description
g	Represents a good state of the system.
f	Represents a failed state of the system.
t	Time variable.
s	Laplace transformation variable.
$P_i(t)$	The probability of the system being in the ith state.
$P_i(s)$	Laplace transformation of $P_i(t)$.
$\lambda_i; 1 \leq i \leq 4$	Failure rate of ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires.
$\mu_i; 1 \leq i \leq 4$	Repair rate of ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires.
S_0	The system is in a good working condition.
S_1	The system is in a failed state due to the failure of the ESP32 Module.
S_2	The system is in a failed state due to the failure of the ultrasonic sensor.
S_3	The system is in a failed state due to the failure of the battery.
S_4	The system is in a failed state due to the failure of the jumper wire.

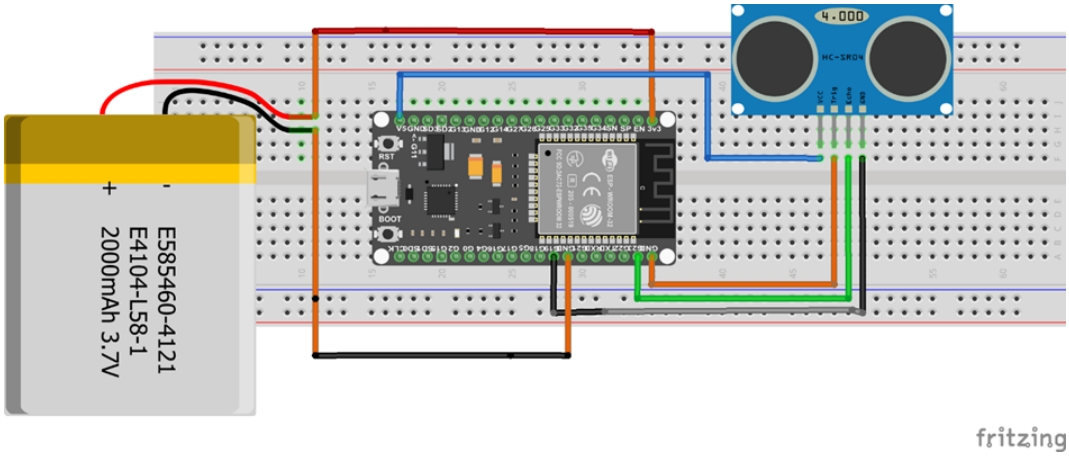

Figure 1. The system electronic scheme

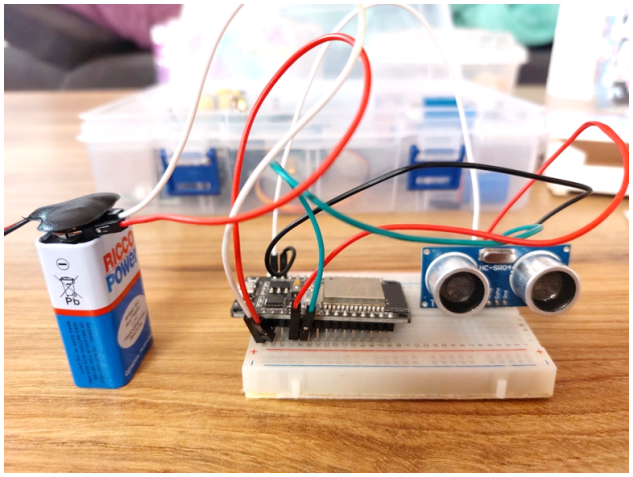

Figure 2. The system prototype

3.2. Mathematical model

The Markov model is widely used in engineering and real-world scenarios due to its state-based modeling approach. Its main advantage is that it depends only on the current system state for transitioning to the next states, disregarding past states. This characteristic is why it's known as a "memoryless model." As a result, the Markov model is crucial for calculating various reliability measures for a system. In the state transition diagram (Figure 1 and Figure 2), the system can be in any of the states S_i, $0 \leq i \leq 4$, at any time t. State S_0 represents the system functioning at full capacity, while states (S_1, S_2, S_3, S_4) indicate system failure states. When considering the system's transition at time $t + \Delta t$, with $\Delta t \to 0$, one can obtain the following Kolmogorov-Chapman differential equations.

$$\left[\frac{d}{dt} + \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4\right] P_0(t) = \mu_1 P_1(t) + \mu_2 P_2(t) + \mu_3 P_3(t) + \mu_4 P_4(t) \tag{1}$$

$$\left[\frac{d}{dt} + \mu_1\right] P_1(t) = \lambda_1 P_0(t) \tag{2}$$

$$\left[\frac{d}{dt} + \mu_2\right] P_2(t) = \lambda_2 P_0(t) \tag{3}$$

$$\left[\frac{d}{dt}+\mu_3\right]P_3(t)=\lambda_3 P_0(t) \tag{4}$$

$$\left[\frac{d}{dt}+\mu_4\right]P_4(t)=\lambda_4 P_0(t) \tag{5}$$

With initial condition;

$$P_i(0)=\begin{cases}1 & \text{if } i\neq 0\\ 0 & \text{if } i=0\end{cases} \tag{6}$$

On taking Laplace transformation of equations from (1) to (5), we get;

$$[s+\lambda_1+\lambda_2+\lambda_3+\lambda_4]\,\bar{P}_0(s)=\mu_1\bar{P}_1(s)+\mu_2\bar{P}_2(s)+\mu_3\bar{P}_3(s)+\mu_4\bar{P}_4(s) \tag{7}$$

$$[s+\mu_1]\,\bar{P}_1(s)=\lambda_1\bar{P}_0(s) \tag{8}$$

$$[s+\mu_2]\,\bar{P}_2(s)=\lambda_2\bar{P}_0(s) \tag{9}$$

$$[s+\mu_3]\,\bar{P}_3(s)=\lambda_3\bar{P}_0(s) \tag{10}$$

$$[s+\mu_4]\,\bar{P}_4(s)=\lambda_4\bar{P}_0(s) \tag{11}$$

On solving equations from (7) to (11), we get,

$$\bar{P}_0(s)=\frac{1}{T_0} \tag{12}$$

$$\bar{P}_i(s)=\frac{1}{T_0}\left(\frac{\lambda_i}{s+\mu_i}\right),\quad 1\leq i\leq 4 \tag{13}$$

Where,

$$T_0=\left[s+\lambda_1+\lambda_2+\lambda_3+\lambda_4-\frac{\mu_1\lambda_1}{s+\mu_1}-\frac{\mu_2\lambda_2}{s+\mu_2}-\frac{\mu_3\lambda_3}{s+\mu_3}-\frac{\mu_4\lambda_4}{s+\mu_4}\right]$$

The system's "upstate" refers to the total states where the system operates with full or partial working capacity, while the "downstate" includes all states where the system completely fails. Therefore, the probabilities of the system being in the upstate or downstate are determined by the following equations.

$$\bar{P}_{\text{up}}(s)=\bar{P}_0(s) \tag{14}$$

$$\bar{P}_{\text{down}}(s)=\bar{P}_1(s)+\bar{P}_2(s)+\bar{P}_3(s)+\bar{P}_4(s) \tag{15}$$

$$\bar{P}_{\text{up}}(s)+\bar{P}_{\text{down}}(s)=\frac{1}{s} \tag{16}$$

Markov modeling:

Markov models are widely used in reliability analysis and various engineering applications due to their ability to model systems with probabilistic transitions between different states. The key features of Markov models include:

- Memoryless Property: A Markov model is characterized by the Markov property, which states that the probability of transitioning to the next state depends only on the current state and not on the sequence of states that preceded it. This ”memoryless” property simplifies the modeling of systems where the future state is conditionally independent of past states.
- State Space Representation: The system is represented by a set of discrete states, where each state corresponds to a specific condition or configuration of the system. Transitions between states occur with certain probabilities, which are typically represented by transition rates or probabilities.
- Transition Rates and Probabilities: In a continuous-time Markov model, the transition rates between states are governed by the rates at which events occur. For example, in our system, transition rates are defined for component failures and repairs. These rates are used to describe the dynamics of the system and calculate various reliability metrics.

Reliability analysis:

Reliability analysis using Markov models involves calculating the probability of the system being in a particular state over time and evaluating its overall performance. Key components of this analysis include:

- Kolmogorov-Chapman Differential Equations: These equations describe the rate of change of the probability of the system being in each state. They are derived from the Markov property and are used to compute the time-dependent probabilities of the system being in different states.
- Laplace Transformations: To solve the differential equations, we use Laplace transformations, which convert the time-domain equations into the frequency domain. This approach simplifies the calculations and allows for the determination of steady-state probabilities and other performance metrics.
- Performance Metrics: Reliability metrics such as mean time to failure (MTTF), system availability, and downtime are derived from the solution of the Markov model. These metrics provide insights into the system’s performance and help assess its reliability under different conditions.

Assumptions of the Markov Model:

The Markov model relies on several assumptions:

- Independent Failures: Initially, we assume that component failures are independent. However, in the revised model, we relax this assumption to consider correlated failures, which may occur in real-world scenarios.
- Exponential Failure and Repair Rates: The model assumes that failure and repair times follow an exponential distribution. This assumption simplifies the analysis but may not fully capture the variability in real-world failure and repair processes. The updated model introduces variability to better reflect real conditions.
- Constant Repair Facilities: The original model assumes constant repair rates, but this has been revised to account for variable repair facilities and logistical constraints.

The expanded theoretical foundation provides a more comprehensive understanding of the Markov modeling approach and its application in reliability analysis. This detailed discussion enhances the manuscript’s robustness and offers a clearer rationale for the modeling choices and assumptions.

3.3. Performance analysis of the smart inventory management system

To assess the reliability of the smart inventory system, we will utilize a table containing component failure rates. This data has been compiled from literature sources and consultations with industry experts. Below, you’ll find the potential failure rates for the components of the inventory system outlined in Table 3. [8];[9]

3.4. Reliability

System reliability refers to the likelihood of the system successfully performing its intended functions over a specified duration and under predefined conditions. Mathematically, reliability can be expressed as the probability that the system will not fail before the specified time period ’t’.

Table 3. Potential failure rates of components in the smart inventory system

Component	(1) Average Failure Years	Failure Rate	(2) Average Failure Years	Failure Rate	(3) Average Failure Years	Failure Rate
ESP32 Module	5.70	0.17	6.70	0.15	7.70	0.13
HC-SR04 Ultrasonic sensor	3.57	0.28	4.57	0.22	5.57	0.18
Battery	0.45	2.21	1.45	0.69	2.45	0.40
Jumper wire	2.50	0.40	3.50	0.28	4.50	0.21

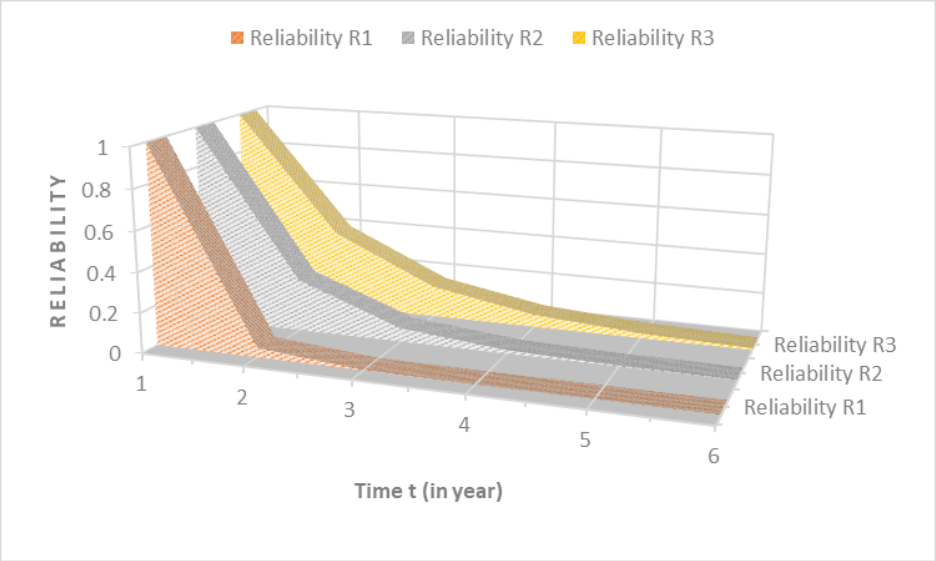

Figure 3. Reliability graph of R_1, R_2 and R_3

$$R(t) = P(T > t) \tag{17}$$

For the calculation of reliability, we set all repair rates to zero in equation (14) and take the inverse Laplace transformation. This allows us to obtain an explicit expression for reliability as follows:

$$R(t) = e^{-(\lambda_1+\lambda_2+\lambda_3+\lambda_4)t} \tag{18}$$

By using the failure rate values provided in Table 3, the reliability functions $R_1(t)$, $R_2(t)$, and $R_3(t)$ of the smart inventory system can be calculated. By varying the time t from 0 to 5, we can generate Table 4 and Figure 3, which illustrate the system's reliability over time.

Table 4. Reliability R_1, R_2 and R_3 of the system

Time t (in years)	$R_1(t) = e^{-3.06t}$	$R_2(t) = e^{-1.34t}$	$R_3(t) = e^{-0.92t}$
0	1.00000	1.00000	1.00000
1	0.04689	0.26185	0.39852
2	0.00220	0.06856	0.15882
3	0.00010	0.01795	0.06329
4	0.00000	0.00470	0.02522
5	0.00000	0.00123	0.01005

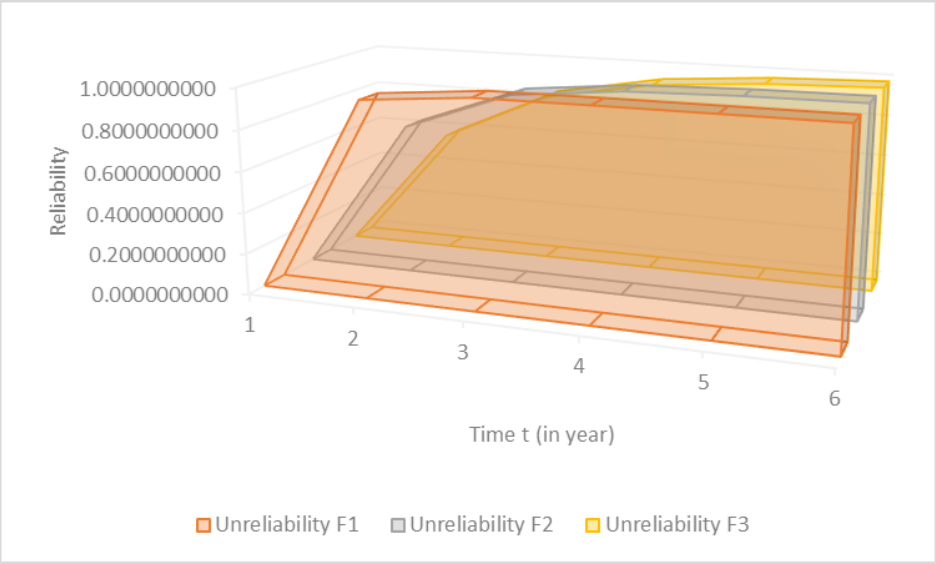

Figure 4. Unreliability graph of F_1, F_2 and F_3

3.5. Unreliability

Unreliability is the probability that a system will fail before a specified time period t is displayed in Table 5 and Figure 4. Mathematically, the unreliability of the system can be expressed as:

$$F(t) = 1 - R(t) \tag{19}$$

Table 5. Unreliability F_1, F_2 and F_3 of the system

Time t **(in years)**	$F_1(t) = 1 - e^{-3.06t}$	$F_2(t) = 1 - e^{-1.34t}$	$F_3(t) = 1 - e^{-0.92t}$
0	0.00000	0.00000	0.00000
1	0.95311	0.73815	0.60148
2	0.99780	0.93144	0.84118
3	0.99990	0.98205	0.93671
4	1.00000	0.99530	0.97478
5	1.00000	0.99877	0.98995

3.6. MTTF

The Mean Time To Failure (MTTF) of the system is defined as the average time until the system experiences its first failure. From an organizational perspective, the MTTF value should be as large as possible. Mathematically, the MTTF is given by:

$$\text{MTTF} = \int_0^\infty R(t)\, dt \tag{20}$$

To determine the Mean Time to Failure (MTTF) of the system, set all repair rates to zero and let s approach zero in $\overline{P}_{\text{up}}(s)$. The MTTF can then be calculated using the formula:

$$\text{MTTF} = \frac{1}{\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4} \tag{21}$$

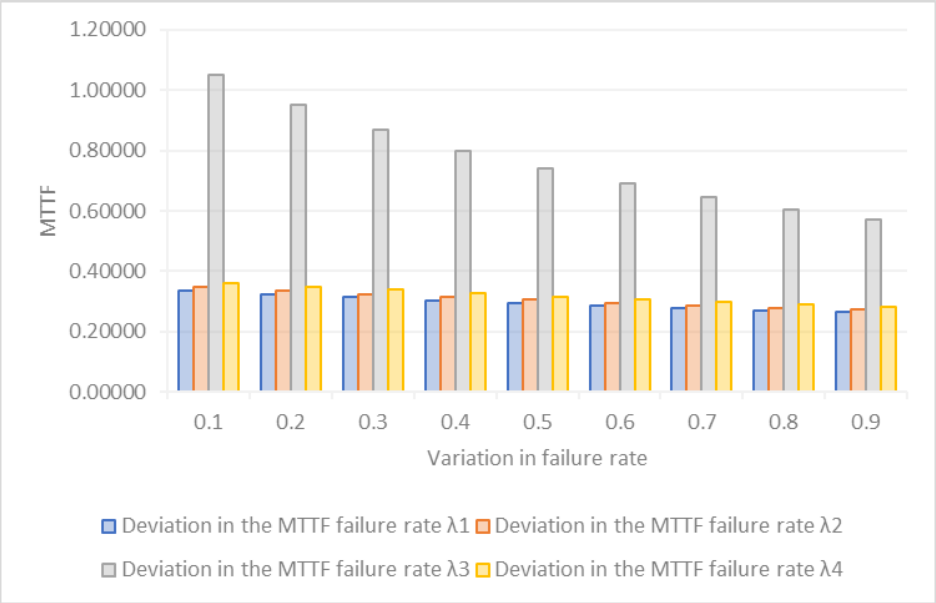

Figure 5. MTTF of the system

where $\lambda_1, \lambda_2, \lambda_3$, and λ_4 are the failure rates of the components.

Initially, set the failure rates to $\lambda_1 = 0.17$, $\lambda_2 = 0.28$, $\lambda_3 = 2.21$, and $\lambda_4 = 0.40$ in Equation (21). Then, vary each failure rate individually from 0.1 to 0.9 while keeping the others constant. The results of these variations are shown in Table 6 and Figure 5.

Table 6. MTTF of the system with reference to variation in failure rates.

Variation in Failure Rate	Deviation in MTTF for λ_1	Deviation in MTTF for λ_2	Deviation in MTTF for λ_3	Deviation in MTTF for λ_4
0.1	0.33445	0.34722	1.05263	0.36232
0.2	0.32362	0.33557	0.95238	0.34965
0.3	0.31348	0.32468	0.86957	0.33784
0.4	0.30395	0.31447	0.80000	0.32680
0.5	0.29499	0.30488	0.74074	0.31646
0.6	0.28653	0.29586	0.68966	0.30675
0.7	0.27855	0.28736	0.64516	0.29762
0.8	0.27100	0.27933	0.60606	0.28902
0.9	0.26385	0.27174	0.57143	0.28090

3.7. Sensitivity analysis of MTTF and Reliability

Understanding the most critical components of the system is crucial. In this study, the authors conduct a sensitivity analysis of the MTTF with respect to variations in failure rates and a sensitivity analysis of reliability with respect to time. This analysis identifies the components that significantly impact the system's MTTF and reliability, highlighting those whose performance is drastically affected by small changes in their failure rates.

To perform the sensitivity analysis of the system's Mean Time to Failure (MTTF), differentiate Equation (21) with respect to λ_i, where $1 \leq i \leq 4$. Set the failure rates to $\lambda_1 = 0.17$, $\lambda_2 = 0.28$, $\lambda_3 = 2.21$, and $\lambda_4 = 0.40$. Then, vary each failure rate individually from 0.1 to 0.9 while keeping the others constant. The results are presented in Table 7 and Figure 6.

The same for sensitivity of reliability, the results are presented in Table 8 and Figure 7.

Table 7. Sensitivity of MTTF of the system

Time t (In year)	$\frac{\partial(\text{MTTF})}{\partial\lambda_1}$	$\frac{\partial(\text{MTTF})}{\partial\lambda_2}$	$\frac{\partial(\text{MTTF})}{\partial\lambda_3}$	$\frac{\partial(\text{MTTF})}{\partial\lambda_4}$
0.1	-0.11186	-0.12056	-1.10803	-0.13127
0.2	-0.10473	-0.11261	-0.90703	-0.12226
0.3	-0.09827	-0.10541	-0.75614	-0.11413
0.4	-0.09239	-0.09889	-0.64000	-0.10680
0.5	-0.08702	-0.09295	-0.54870	-0.10014
0.6	-0.08210	-0.08753	-0.47562	-0.09409
0.7	-0.07759	-0.08257	-0.41623	-0.08858
0.8	-0.07344	-0.07803	-0.36731	-0.08353
0.9	-0.06962	-0.07384	-0.32653	-0.07890

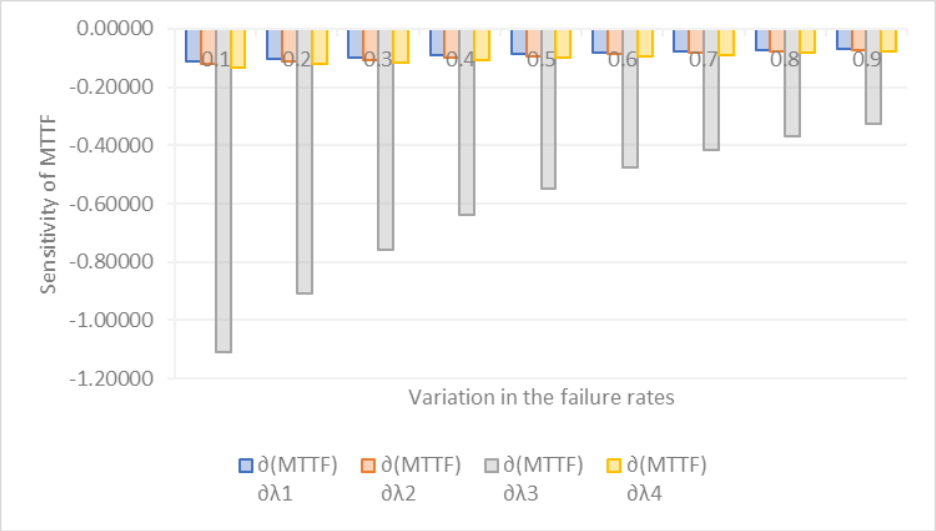

Figure 6. Sensitivity of MTTF of the system with variation in failure rates

Table 8. Sensitivity of reliability of the system

Time t (In year)	$\frac{\partial R(t)}{\partial\lambda_1}$	$\frac{\partial R(t)}{\partial\lambda_2}$	$\frac{\partial R(t)}{\partial\lambda_3}$	$\frac{\partial R(t)}{\partial\lambda_4}$
0	0.0000000000	0.0000000000	0.0000000000	0.0000000000
1	-0.0468876952	-0.0468876952	-0.0468876952	-0.0468876952
2	-0.0043969119	-0.0043969119	-0.0043969119	-0.0043969119
3	-0.0003092416	-0.0003092416	-0.0003092416	-0.0003092416
4	-0.0000193328	-0.0000193328	-0.0000193328	-0.0000193328
5	-0.0000011331	-0.0000011331	-0.0000011331	-0.0000011331
6	-0.0000000638	-0.0000000638	-0.0000000638	-0.0000000638
7	-0.0000000035	-0.0000000035	-0.0000000035	-0.0000000035
8	-0.0000000002	-0.0000000002	-0.0000000002	-0.0000000002
9	0.0000000000	0.0000000000	0.0000000000	0.0000000000

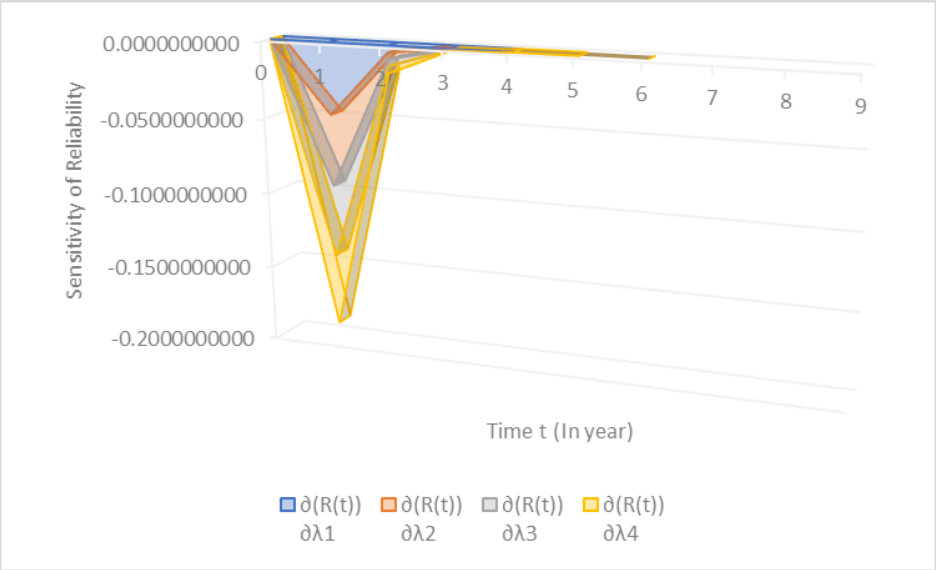

Figure 7. Sensitivity of reliability of the system with variation in failure rates

3.8. Availability

Availability is the probability that a system is operational and able to perform its intended function at any given point in time. It is a key measure of system maintainability and reliability. Mathematically, availability can be expressed as:

$$A(t) = \frac{\text{MTBF}}{\text{MTBF} + \text{MTTR}} \tag{22}$$

where MTBF (Mean Time Between Failures) represents the average time between failures of the system, and MTTR (Mean Time to Repair) denotes the average time required to repair the system after a failure. This formula highlights that availability depends on both the reliability of the system (as indicated by MTBF) and the effectiveness of maintenance procedures (as reflected by MTTR).

Table 9. MTBF and Availability of the system components

Component	**Failure rate**	**MTBF (Years)**	**MTTR (Years)**	**Availability (%)**
ESP32 Module (1)	0.17	5.88	0.5	92.18
ESP32 Module (2)	0.15	6.67	0.5	93.00
ESP32 Module (3)	0.13	7.69	0.5	93.92
HC-SR04 Ultrasonic sensor (1)	0.28	3.57	0.5	87.71
HC-SR04 Ultrasonic sensor (2)	0.22	4.55	0.5	90.11
HC-SR04 Ultrasonic sensor (3)	0.18	5.56	0.5	91.80
Battery (1)	2.21	0.45	0.5	47.39
Battery (2)	0.69	1.45	0.5	74.39
Battery (3)	0.40	2.50	0.5	83.33
Jumper wire (1)	0.40	2.50	0.5	83.33
Jumper wire (2)	0.28	3.57	0.5	87.71
Jumper wire (3)	0.21	0.76	0.5	90.48

Table 9 provides a comprehensive overview of the reliability and availability metrics for various components used in the system. It highlights that the ESP32 Modules exhibit relatively high availability, with values ranging from 92.18% to 93.92%, attributed to their low failure rates and longer Mean Time Between Failures (MTBF). In contrast, the Battery components display the lowest availability, especially Battery (1) with only 47.39%, which is a significant outlier due to its high failure rate and short MTBF. The Jumper wires and HC-SR04 Ultrasonic sensors fall in between, showing moderate availability, with improvements observed in the latter iterations of these components. Overall, this data underscores the importance of component selection and optimization in enhancing the system's overall reliability and availability.

3.9. Maintainability

Maintainability measures how easily and quickly a system can be repaired after a failure. It can be expressed as:

$$M = \frac{1}{\text{MTTR}} \tag{23}$$

where MTTR stands for Mean Time To Repair.

Here, a lower MTTR indicates higher maintainability. The calculation for MTTR can be based on industry standards or expert consultation.

Table 10. Maintainability

Component	**MTTR (Years)**	**Maintainability (Repairs/Year)**
All components	0.5	2

3.10. Overall System Efficiency OSE

This metric considers both the performance and reliability of the system. It can be quantified by combining availability with other performance indicators, such as throughput or processing speed.

$$\text{OSE} = \text{Availability} \times \text{Performance Index} \tag{24}$$

Where the Performance Index could be a function of system responsiveness, accuracy, and other relevant factors. This would provide a holistic view of how well the system performs under typical operating conditions.

Table 11. Overall System Efficiency

Component	**Availability (%)**	**Performance Index**	**Overall System Efficiency (%)**
ESP32 Module (1)	92.18	1	92.18
ESP32 Module (2)	93	1	93
ESP32 Module (3)	93.92	1	93.92
HC-SR04 Ultrasonic sensor (1)	87.71	1	87.71
HC-SR04 Ultrasonic sensor (2)	90.11	1	90.11
HC-SR04 Ultrasonic sensor (3)	91.8	1	91.8
Battery (1)	47.39	1	47.39
Battery (2)	74.39	1	74.39
Battery (3)	83.33	1	83.33
Jumper wire (1)	83.33	1	83.33
Jumper wire (2)	87.71	1	87.71
Jumper wire (3)	90.48	1	90.48

3.11. Smart contract deployment

In our study of smart inventory management systems, we explored the deployment of blockchain technology to enhance security, transparency, and efficiency. Blockchain, renowned for its decentralized and immutable ledger capabilities, has garnered widespread recognition across industries. Here, we detail the process of deploying blockchain within our system, which relies on hardware components including the ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires. We initiated our blockchain deployment by focusing on the implementation of smart contracts—self-executing, code-driven protocols executed on a blockchain network. These contracts enable automated, trustless interactions among stakeholders and data sources, enhancing security through cryptographic measures and fostering transparency via decentralization. Our objective was to develop a Solidity smart contract tailored to our smart inventory system, facilitating the secure recording of sensor data on the blockchain. The development process involved the creation of a Solidity file in Remix Ethereum—a web-based integrated development environment (IDE). Within this file, named "InventoryStatus" we defined a struct to store sensor data and implemented functions for device authorization, data addition, and retrieval. Through rigorous testing and refinement, we ensured the reliability and integrity of the contract code. Once the contract code was finalized, we proceeded to compile it using Remix Ethereum, selecting the appropriate Solidity version to generate bytecode and ABI. Subsequently, we deployed the contract onto the Ethereum blockchain using Remix's "Deploy and Run Transactions" tab, with "Injected Web3" facilitating connectivity to our Metamask digital wallet. Figure 8 displays the remix Ethereum platform of our system. The developed solidity code is as follows:

```
// SPDX-License-Identifier: MIT
pragma solidity \^0.8.0;
 contract InventoryStatus \{
enum ComponentState \{ Working, Failed \}
 ComponentState constant DEFAULT\_STATE = ComponentState.Working;
 // Hardware components
ComponentState public esp32State;
ComponentState public ultrasonicSensorState;
ComponentState public batteryState;
ComponentState public jumperWiresState;
 // Events
event ComponentFailure(string componentName);
 constructor() \{
esp32State = DEFAULT\_STATE;
ultrasonicSensorState = DEFAULT\_STATE;
batteryState = DEFAULT\_STATE;
jumperWiresState = DEFAULT\_STATE;
\}
 // Function to simulate failure of a component
function failComponent(string memory componentName) external \{
require(esp32State != ComponentState.Failed, "System already in a
failed state");
 if (keccak256(abi.encodePacked(componentName))
 == keccak256(abi.encodePacked("ESP32"))) \{
esp32State = ComponentState.Failed;
emit ComponentFailure("ESP32");
\} else if (keccak256
(abi.encodePacked(componentName))
==
keccak256(abi.encodePacked("UltrasonicSensor"))) \{
ultrasonicSensorState = ComponentState.Failed;
```

```
emit ComponentFailure("UltrasonicSensor");
\} else if (keccak256(abi.encodePacked(componentName))
== keccak256(abi.encodePacked("Battery"))) \{
batteryState = ComponentState.Failed;
emit ComponentFailure("Battery");
\} else if (keccak256(abi.encodePacked(componentName))
== keccak256(abi.encodePacked("JumperWires"))) \{
jumperWiresState = ComponentState.Failed;
emit ComponentFailure("JumperWires");
\}
\}
 // Function to repair a failed component
function repairComponent(string memory componentName) external \{

require(esp32State == ComponentState.Failed, "System is not in a failed
state");
 if (keccak256(abi.encodePacked(componentName)) == keccak256
 (abi.encodePacked("ESP32"))) \{
esp32State = ComponentState.Working;
\} else if (keccak256(abi.encodePacked(componentName))
== keccak256(abi.encodePacked("UltrasonicSensor"))) \{
ultrasonicSensorState = ComponentState.Working;
\} else if (keccak256(abi.encodePacked(componentName))
== keccak256(abi.encodePacked("Battery"))) \{
batteryState = ComponentState.Working;
\} else if (keccak256(abi.encodePacked(componentName))
== keccak256(abi.encodePacked("JumperWires"))) \{
jumperWiresState = ComponentState.Working;
\}
\}
\}
```

During the deployment process, we unlocked and connected Metamask to Remix, specified initial parameters for the contract, and initiated the deployment transaction. Metamask prompted us to confirm the transaction, which we reviewed and confirmed after ensuring that our Sepolia Faucets test tokens covered the associated gas fees. Upon successful deployment, Remix provided us with the contract address, which we also verified in our Metamask transaction history. With the contract deployed, we proceeded to interact with it using Remix's intuitive interface, executing functions such as device authorization, data addition, and retrieval. Blockchain Platform: The smart inventory management system utilizes the Ethereum blockchain platform. Ethereum is selected due to its robust support for smart contracts and its extensive developer ecosystem. The platform's decentralized nature ensures that inventory records are immutable and transparently managed, aligning with the system's goals of enhanced data security and integrity. Consensus Mechanism: Ethereum employs a Proof of Stake (PoS) consensus mechanism. PoS was chosen for its efficiency and scalability benefits compared to Proof of Work (PoW). Under PoS, validators are selected to create new blocks based on the amount of cryptocurrency they hold and are willing to "stake" as collateral. This mechanism reduces the computational overhead and energy consumption associated with block validation, which is particularly beneficial for maintaining an efficient and scalable inventory management system. Smart Contract Design: The smart contracts in our system are designed to automate and enforce inventory transactions and management processes. Key features of the smart contracts include:

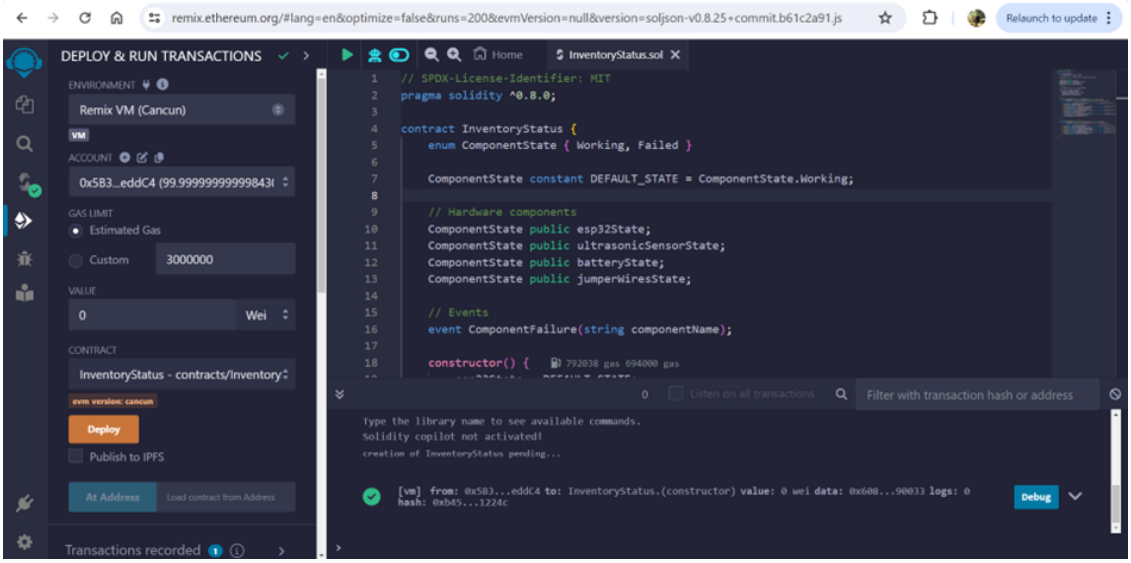

Figure 8. The deployed contract on Remix.Ethereum

- Inventory Tracking: Smart contracts handle the recording and verification of inventory transactions, including additions, removals, and adjustments. Each transaction is logged on the blockchain, ensuring that inventory records are accurate and tamper-proof.
- Access Control: Smart contracts enforce access control policies by defining and managing user permissions. Only authorized personnel can perform certain actions, such as modifying inventory records or initiating transactions, thereby enhancing security and preventing unauthorized access.
- Automated Alerts: The smart contracts include logic to generate automated alerts based on predefined conditions, such as low stock levels or system anomalies. These alerts are issued to relevant stakeholders, ensuring timely responses to potential issues.
- Audit Trails: Smart contracts maintain a complete audit trail of all inventory-related activities. This trail includes details of each transaction, including timestamps, involved parties, and transaction details, providing transparency and accountability.

By integrating these blockchain elements, the smart inventory management system achieves a high level of security, transparency, and automation. The detailed implementation enhances the system's practical applicability and robustness in real-world scenarios. Throughout the deployment and interaction process, we maintained a vigilant eye on gas fees, ensuring that our test token balance in Metamask was sufficient to cover transaction costs. By meticulously following these steps, we successfully implemented and deployed our Solidity smart contract, showcasing the seamless integration of blockchain technology into our smart inventory management system.

4. Results and discussion

In this study, a mathematical model was developed using the Markov model for a smart inventory management system, leveraging IoT principles. The system comprises four key hardware components: ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires. Initially, all components are in optimal working condition, transitioning to a failed state if any component malfunctions. Repairs or replacements are conducted on failed components to restore functionality. Component failure rates are assumed to follow an exponential distribution. Kolmogorov differential equations were derived from the system's transition state diagram and solved using Laplace transformation. This facilitated the derivation of explicit expressions for the system's reliability,

unreliability, and MTTF across various combinations of failure rates. Sensitivity analyses of MTTF and reliability were performed to identify critical components. Key findings include:

- Table 4 and Figure 3 depict system reliabilities for different combinations of failure rates. Higher failure rates significantly diminish system reliability, with Reliability R1 dropping to 4.69% after 1 year. Introducing redundant components and employing higher-quality parts may bolster system reliability.
- Table 5 and Figure 4 illustrate system unreliability, with Unreliability F1 reaching 95.31% after 1 year. Enhancements in component quality and redundancy could lead to improved system reliability.
- Table 6 and Figure 5 present the system's MTTF, indicating a low sensitivity to switch failure rates but a high sensitivity to battery failure rates.
- Table 7 and Figure 6 demonstrate the MTTF's sensitivity to component failure rates, highlighting the substantial impact of servo motor failure rates on MTTF.
- Table 8 and Figure 7 outline system reliability sensitivity, revealing consistent sensitivity across all components. Over 5 years, system reliability stabilizes and remains nearly constant.

To assess the impact of integrating blockchain technology and compare our system with existing traceability systems, Table 12 presents a comparative overview.

Table 12. Comparative analysis with other existing systems

Comparative Metric	**Our IoT System (with Blockchain)**	**System B**	**System C**
Technology	Blockchain (Ethereum)	Blockchain (Hyperledger Fabric)	Non-Blockchain (Wireless Sensor Network & Cloud)
Strengths	Enhanced data security and transparency; Tamper-proof inventory records; Secure traceability throughout supply chain	Secure and verifiable data throughout the food supply chain; Enhanced transparency for consumers	Cost-effective data collection with sensor networks; Real-time inventory monitoring with cloud; Scalable sensor network deployment (potentially)
Weaknesses	Minor performance overhead; Gas fees associated with blockchain transactions	Limited scalability compared to public blockchains; Complex network setup and management	Potential security vulnerabilities in cloud storage; Reliance on internet connectivity for cloud access
Applications	Ideal for applications requiring high data security and tamper-proof records (e.g., pharmaceuticals, high-value goods); Temperature and humidity monitoring for perishable goods	Food traceability for improved food safety and quality control	Precision agriculture and environmental monitoring in remote locations
Cost Considerations	Affordable sensors may offset blockchain complexity	Potentially higher costs due to platform setup and maintenance	Lower sensor costs, but potential cloud storage fees and internet connectivity costs
Focus	Specific focus on secure and transparent temperature & humidity data management	Secure and efficient food traceability with a focus on smart agriculture	Environmental monitoring with a focus on optimizing agricultural practices
Scalability	Potentially more scalable with a public blockchain (Ethereum)	Limited scalability due to the consortium model (Hyperledger Fabric)	Potentially scalable sensor network deployment, but cloud storage limitations exist

In the pursuit of enhancing the security, transparency, and efficiency of supply chain management, various IoT-based traceability systems have emerged, each leveraging different technologies and approaches. To provide a comprehensive assessment, the following comparative analysis examines our IoT system integrated with blockchain technology against two other existing systems: System B [7], which employs Hyperledger Fabric, and System C [10], which relies on traditional wireless sensor networks and cloud storage.

Technology

- Our IoT System (with Blockchain - Ethereum): The integration of Ethereum blockchain in our system offers decentralized security, making it ideal for applications requiring immutable and transparent records. Ethereum's public blockchain nature ensures that data is tamper-proof, a critical feature for high-value goods and sensitive environments.
- System B (Blockchain - Hyperledger Fabric): System B utilizes Hyperledger Fabric, a permissioned blockchain, which provides secure and verifiable data transactions within a controlled consortium. While it excels in environments where data privacy is paramount, its scalability is limited compared to public blockchains like Ethereum.
- System C (Non-Blockchain - Wireless Sensor Network and Cloud): System C's reliance on a wireless sensor network combined with cloud storage offers a cost-effective solution for real-time monitoring. However, it lacks the inherent security and immutability features provided by blockchain, making it vulnerable to data tampering and breaches.

Strengths

- Our IoT System: The key strength of our system lies in its enhanced data security and transparency, driven by blockchain's decentralized nature. The immutability of inventory records ensures tamper-proof traceability throughout the supply chain, making it a robust solution for industries requiring stringent data integrity.
- System B: System B offers secure and verifiable data across the food supply chain, with a focus on transparency that benefits both suppliers and consumers. The use of Hyperledger Fabric ensures that data remains controlled and auditable within the network.
- System C: System C excels in cost-effective data collection through sensor networks, offering real-time inventory monitoring via cloud services. Its scalable sensor deployment potential is advantageous for large-scale operations, though it does not inherently address security concerns.

Weaknesses

- Our IoT System: While blockchain integration enhances security, it introduces a minor performance overhead and incurs gas fees associated with transactions on the Ethereum network. These factors could affect the system's efficiency, particularly in high-frequency transaction environments.
- System B: System B's use of a permissioned blockchain limits its scalability compared to public blockchain solutions. Additionally, the complexity involved in setting up and managing the Hyperledger Fabric network can be a barrier to adoption.
- System C: The reliance on cloud storage presents potential security vulnerabilities, as cloud environments are often targeted by cyber threats. Furthermore, the dependency on internet connectivity for cloud access could pose challenges in remote or underdeveloped regions.

Applications

- Our IoT System: The system is particularly suited for applications demanding high data security and tamper-proof records, such as pharmaceuticals and high-value goods. It is also tailored for monitoring temperature and humidity, crucial for managing perishable goods.
- System B: System B is optimized for food traceability, ensuring food safety and quality control through secure and efficient data management. Its application is most effective in smart agriculture and other sectors where controlled data sharing is essential.
- System C: System C's application shines in precision agriculture and environmental monitoring, especially in remote locations where cost-effective sensor deployment is critical. However, its reliance on cloud infrastructure may limit its effectiveness in areas with poor connectivity.

Cost Considerations

- Our IoT System: The system's affordability is enhanced by the use of cost-effective sensors, which helps offset the complexity introduced by blockchain integration. However, the gas fees associated with Ethereum transactions must be considered in the overall cost assessment.

- System B: System B may incur higher costs due to the initial setup and maintenance of the Hyperledger Fabric platform. These costs could be justified in scenarios where the value of secure and private data management outweighs the expenses.
- System C: While System C benefits from lower sensor costs, it faces potential additional costs related to cloud storage fees and internet connectivity. These costs could accumulate over time, particularly in large-scale deployments.

Focus

- Our IoT System: The primary focus of our system is on the secure and transparent management of temperature and humidity data, which is critical for preserving the integrity of perishable goods throughout the supply chain.
- System B: System B is focused on secure and efficient food traceability, with an emphasis on enhancing the safety and quality of agricultural products. Its application in smart agriculture makes it a valuable tool for improving food supply chains.
- System C: System C is concentrated on environmental monitoring, with a particular focus on optimizing agricultural practices. Its ability to provide real-time data in remote areas is a key advantage, although it lacks the security features of blockchain-based systems.

Scalability

- Our IoT System: The scalability of our system is potentially enhanced by the use of a public blockchain like Ethereum, which can support a large number of transactions across a distributed network. However, scalability may be influenced by the performance overhead and transaction fees.
- System B: System B's scalability is constrained by the consortium model of Hyperledger Fabric, which may limit its expansion beyond a certain number of participants. Despite this, it remains effective in controlled environments where privacy and security are prioritized.
- System C: System C offers the potential for scalable sensor network deployment, but its scalability is ultimately limited by the capacity and security of the cloud storage infrastructure. This limitation could hinder its applicability in very large-scale operations.

In summary, each system presents unique strengths and weaknesses, making them suitable for different applications within supply chain management. Our IoT system, with its blockchain integration, offers unparalleled security and transparency, making it ideal for industries where data integrity is paramount. System B provides a secure solution for food traceability, though its scalability is limited. System C offers cost-effective and scalable monitoring but lacks the security guarantees of blockchain. The choice of system should be guided by the specific requirements of the application, including security needs, scalability, and cost considerations.

5. Discussion on Generalizability

5.1. Generalizability to Other Hardware Configurations

The proposed model is designed with flexibility to accommodate a variety of hardware configurations beyond the specific components used in our smart inventory management system. By adjusting the failure and repair rates to reflect different IoT devices and sensors, the model can be adapted to diverse hardware setups. For example, if the system is implemented with alternative sensors or modules, the parameters for failure rates and repair strategies should be recalibrated accordingly. This adaptability ensures that the core principles of the model remain effective across different hardware configurations, allowing for its application in various technological contexts.

5.2. Applicability to Different Industries

The principles of the proposed model extend beyond inventory management and can be applied to a wide range of industries. In healthcare, for instance, the model can be adapted to monitor medical equipment and ensure

reliability in critical environments. In agriculture, it can be used to track environmental conditions and equipment performance. For manufacturing, the model can assist in monitoring machinery and production lines. Each industry may require specific adaptations, such as integrating different types of sensors or adjusting for industry-specific operational conditions. However, the foundational approach of using Markov modeling for reliability analysis and incorporating blockchain for data integrity remains applicable.

5.3. Potential Adaptations Needed

To generalize the model for various contexts, certain adaptations may be necessary. These include modifying the model to account for different environmental conditions, component specifications, and industry-specific requirements. For instance, in environments with higher humidity or extreme temperatures, adjustments to the failure rates and repair mechanisms might be required. Additionally, industry-specific features such as the type of data collected or the frequency of monitoring may necessitate changes in the model's parameters. The model's inherent flexibility supports these modifications, allowing it to be tailored to meet the needs of diverse applications while maintaining its core functionality.

5.4. Limitations

While the model demonstrates significant flexibility, there are limitations to consider. The generalizability of the model assumes that the underlying assumptions—such as independent component failures and exponential failure and repair rates—hold true across different hardware configurations and industries. In real-world scenarios, these assumptions may not always be accurate. For example, correlated component failures or varying repair times may impact the model's performance and accuracy. Additionally, the effectiveness of the blockchain integration may vary depending on the specific platform and its capabilities, potentially affecting the model's applicability. Future work should address these limitations by exploring more complex failure patterns and conducting empirical validations across diverse settings.

5.5. Future Work

- Extensions to the Model: To enhance the proposed model, several extensions are planned. Future research will explore incorporating more complex failure patterns, such as correlated failures and non-exponential failure and repair rates. These additions aim to provide a more realistic representation of failure dynamics and improve the model's applicability to intricate real-world scenarios. Additionally, integrating predictive maintenance capabilities using machine learning techniques could further advance the model. By analyzing historical data to forecast potential component failures, we can enhance the system's reliability and reduce downtime.
- Further Validation with Additional Datasets: To validate the model's performance and robustness, we propose conducting empirical evaluations using diverse datasets from various industries and hardware configurations. This validation will involve testing the model with real-world data to assess its accuracy and adaptability in different contexts. Additionally, performing sensitivity analyses will help evaluate how variations in parameters and assumptions impact the model's outcomes. Collecting and integrating data from multiple sources will offer a comprehensive understanding of the model's effectiveness and limitations.
- Collaboration with Industry Partners: Engaging with industry partners for collaborative studies is also recommended. This collaboration will provide practical insights and feedback on the model's performance in operational settings. Real-world case studies and pilot projects will generate valuable data to refine and validate the model further.

By pursuing these directions for future work, we aim to improve the model's capabilities and ensure its relevance across a wider range of scenarios. These efforts will contribute to the ongoing development and practical application of the model.

6. Conclusion

The evolution of inventory management has undergone a remarkable transformation in recent years, driven by technological advancements and the quest for heightened efficiency, precision, and security. Traditional inventory management paradigms have gradually yielded ground to smart inventory management systems, leveraging cutting-edge technologies like the Internet of Things (IoT), automation, and data analytics to streamline operations and enrich decision-making processes. However, despite their myriad benefits, these systems also present novel challenges, particularly in terms of reliability, security, and transparency. A paramount challenge confronting smart inventory management systems lies in the reliability of their hardware components. Unlike their conventional counterparts, these systems heavily rely on a spectrum of hardware elements—sensors, actuators, and communication devices—to capture, process, and transmit data in real-time. Any lapse or malfunction in these components can precipitate operational disruptions, leading to inventory discrepancies, delays, and ultimately, financial ramifications. Therefore, ensuring the reliability and availability of these components emerges as a linchpin for the success of smart inventory management systems. In response to this challenge, this study has introduced a novel quantitative mathematical model that scrutinizes the performance of smart inventory management systems through a reliability lens. Central to this model is the inclusion of essential hardware components commonly deployed in such systems, notably the ESP32 module, HC-SR04 ultrasonic sensor, battery, and jumper wires. By harnessing Markov modeling and a reliability-centric approach, our model aims to pinpoint critical components vulnerable to failure and prioritize maintenance interventions accordingly. An integral facet of our methodology involves the integration of blockchain technology, which augments the security and transparency of smart inventory management systems. Blockchain, initially devised as the foundational framework for cryptocurrencies, has garnered considerable acclaim across diverse sectors for its capacity to furnish a secure and immutable ledger of transactions. In the context of inventory management, blockchain offers several advantages, including immutability, decentralization, and transparency. By logging all inventory-related transactions on a distributed ledger, blockchain upholds data integrity and fosters trust among stakeholders, curtailing the risk of fraudulent activities or unauthorized access. Moreover, our research evaluates pivotal reliability metrics such as reliability, unreliability, and Mean Time to Failure (MTTF) to quantify the efficacy of smart inventory management systems. Through sensitivity analysis, we have identified critical components exerting a pronounced influence on system reliability, underscoring the imperative of proactive maintenance strategies. Notably, our findings spotlight components such as switches and servo motors as particularly sensitive to failure, warranting diligent monitoring and maintenance to forestall operational disruptions. In essence, this paper constitutes a substantive contribution to the extant literature on smart inventory management systems by proffering a holistic mathematical model that seamlessly integrates blockchain technology to fortify security and transparency. By accentuating reliability metrics and advocating proactive maintenance protocols, our research endeavors to optimize the performance of smart inventory management systems, safeguarding their reliability and availability amidst the burgeoning landscape of Industry 4.0.

REFERENCES

1. Tássia Bolotari Affonso, Samuel Vieira Conceição, Leandro Reis Muniz, João Flávio de Freitas Almeida, and Juliana Cássia de Lima. A new hybrid forecasting method for spare part inventory management using heuristics and bootstrapping. *Decision Analytics Journal*, 10:100415, 3 2024.
2. Love Allen Chijioke Ahakonye, Ahmad Zainudin, Md Javed Ahmed Shanto, Jae Min Lee, Dong Seong Kim, and Taesoo Jun. A multi-MLP prediction for inventory management in manufacturing execution system. *Internet of Things*, 26:101156, 7 2024.
3. Rajesh Bose, Haraprasad Mondal, Indranil Sarkar, and Sandip Roy. Design of smart inventory management system for construction sector based on IoT and cloud computing. *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, 2:100051, 1 2022.
4. Pietro De Giovanni. Smart Supply Chains with vendor managed inventory, coordination, and environmental performance. *European Journal of Operational Research*, 292(2):515–531, 7 2021.
5. Ameneh Farahani, Ahmad Shoja, and Hamid Tohidi. Markov and semi-Markov models in system reliability. *Engineering Reliability and Risk Assessment*, pages 91–130, 1 2023.
6. Xiaoming Li. Inventory management and information sharing based on blockchain technology. *Computers & Industrial Engineering*, 179:109196, 5 2023.

7. Jun Lin, Zhiqi Shen, Anting Zhang, and Yueting Chai. Blockchain and IoT based Food Traceability for Smart Agriculture. In *Proceedings of the 3rd International Conference on Crowd Science and Engineering*, pages 1–6, New York, NY, USA, 7 2018. ACM.
8. Albert-Ngabo Niyonsenga and Marcelo M. Wanderley. Tools and Techniques for the Maintenance and Support of Digital Musical Instruments. In *Proceedings of the 2023 International Conference on New Interfaces for Musical Expression (NIME2023)*, Mexico City, MX, 2023.
9. Per Olzon and Vanessa Vannas. *Generic Sensor Error Modelling Using Radar Equation for Estimating the Sensor Error in Ultrasonic Sensors*. PhD thesis, Chalmers University of Technology, Gothenburg, 2020.
10. Ahmed Radhi. Design and Implementation of a Smart Farm System. *Association of Arab Universities Journal of Engineering Sciences*, 24(3):227–241, 2017.
11. Esha Saha and Pradeep Rathore. A smart inventory management system with medication demand dependencies in a hospital supply chain: A multi-agent reinforcement learning approach. *Computers & Industrial Engineering*, page 110165, 4 2024.
12. Yasin Tadayonrad and Alassane Balle Ndiaye. A new key performance indicator model for demand forecasting in inventory management considering supply chain reliability and seasonality. *Supply Chain Analytics*, 3:100026, 9 2023.