CHAINGUARD AI: An IoT-Enabled Framework for Cold-Chain Shipment Monitoring, Risk Estimation and Anomaly Detection

Zainab Imran Shaikh[1], Mehak Chainsingh Patel[2], Prof. Bharti Ganesh Salve[3]
[1,2]*Bachelor Students, Department of Artificial Intelligence and Data Science, Shri Chhatrapati Shivaji Maharaj College of Engineering, Ahmednagar, Maharashtra, India*
[3]*Asst. Professor, Department of Artificial Intelligence and Data Science, Shri Chhatrapati Shivaji Maharaj College of Engineering, Ahmednagar, Maharashtra, India*

***Abstract*—Cold-chain logistics requires continuous visibility of environmental conditions because temperature excursions, excessive humidity, transit delays and equipment faults can compromise temperature-sensitive goods. This paper presents CHAINGUARD AI, a prototype framework integrating shipment monitoring, machine-learning-based risk estimation, anomaly detection, rule-based alerts, route comparison and what-if scenario analysis. The supplied software uses a React and TypeScript frontend, a FastAPI backend, SQLAlchemy persistence, a Random Forest classifier and an Isolation Forest detector. The model feature schema includes temperature, relative humidity, transit duration, delivery delay, temperature exposure and recent temperature change.**

A predicted positive-class probability is mapped to a 0–100 prototype risk score and four display bands. Source-code inspection confirms that the current v0.1.0 model labels, seeded shipment records, telemetry and route inputs are synthetic; the supplied archive does not contain ESP32 firmware or a live DHT22 ingestion endpoint. To demonstrate the analytical workflow without presenting fabricated field evidence, a separate reproducible synthetic benchmark was generated using the same six-feature schema and a synthetic event-generation rule.

On a stratified held-out synthetic test partition, the Random Forest achieved 85.60% accuracy, 53.44% recall, 56.45% F1-score and 0.811 ROC-AUC; logistic regression achieved 82.67% accuracy, 61.83% recall, 55.48% F1-score and 0.836 ROC-AUC. These values are simulation-only and cannot be interpreted as real shipment prediction performance. The paper documents the architecture, implementation, synthetic experiment and validation requirements for subsequent real-world testing.

***Index Terms*—Cold-chain logistics, Internet of Things, shipment monitoring, Random Forest, Isolation Forest, anomaly detection, risk assessment, predictive analytics.**

I. INTRODUCTION

Temperature-sensitive products, including pharmaceuticals, vaccines, biological materials and perishable foods, depend on controlled conditions during storage and transportation. A shipment may experience a temperature excursion because of refrigeration failure, handling practices, route disruption, extended delivery time or poor container sealing. Periodic manual inspection can leave gaps between observations, while isolated data loggers may provide limited support for interpreting the combined effect of environmental conditions and transit context. Research on cold-chain tracking has examined how continuous condition monitoring and information sharing can support logistics decisions [1], [2].

IoT sensing can provide repeated temperature and humidity measurements, while analytics can convert time-series observations into risk indicators and alerts. Recent literature highlights the potential of combining IoT data acquisition with AI-based predictive analytics, while identifying integration and operational validation as important research needs [3]. A practical monitoring platform must distinguish a threshold breach from a predictive estimate, preserve reading provenance and communicate uncertainty to operators.

CHAINGUARD AI is a software-first prototype designed to combine these functions. It includes a

shipment control tower, telemetry history, Random Forest risk estimation, Isolation Forest anomaly detection, threshold and trend checks, route comparison, a what-if simulator, alert management and CSV reporting. The supplied project manifest identifies the current mode as simulated telemetry and lists synthetic model data and simulated route data as limitations.

The contributions are: (i) a modular shipment-monitoring and risk-intelligence architecture; (ii) a six-feature risk-estimation workflow combining environmental and transit context; (iii) a hybrid anomaly and alert mechanism; and (iv) a reproducible synthetic experiment and a validation protocol that separates software demonstration from empirical claims. The contribution is the integrated prototype and evaluation design, not a claim that the underlying ML algorithms are new.

II. BACKGROUND AND RELATED WORK

Cold-chain tracking involves collecting condition data across storage, handling and transportation stages and using the information to support product-quality and logistics decisions. Badia-Melis et al. review monitoring technologies and traceability approaches relevant to food supply chains [2].

Chaudhuri et al. discuss how data analytics can support cold-chain logistics decisions and emphasize translating monitoring variables into actionable assessments [1].

The integration of IoT and AI has been studied as a means of improving monitoring, traceability and predictive assessment in cold-chain systems. Müller et al. review sensor-based data acquisition and AI analytics and identify the need for integrated solutions and practical validation [3].

Random Forest aggregates decision trees and is widely used for structured classification tasks [4]. Isolation Forest identifies unusual observations through random partitioning and isolation depth [5]. In CHAINGUARD AI these are prototype components. Their use alone does not establish predictive validity: performance depends on representative data, target-label quality, appropriate data splits and comparison with suitable baselines.

III. RESEARCH PROBLEM, OBJECTIVES AND SCOPE

3.1 Problem statement

Fragmented monitoring workflows may fail to combine environmental telemetry with shipment context in a unified risk-review process. The research problem is to design a prototype that estimates a shipment risk indicator and identifies unusual readings through supervised learning, unsupervised anomaly detection and explicit operating-limit checks.

3.2 Objectives

- Represent each shipment with product category, origin, destination, vehicle and configured environmental limits.
- Store timestamped temperature and humidity observations with shipment identifiers and location metadata.
- Estimate a prototype risk probability using temperature, humidity, transit duration, delay, exposure and temperature-rise features.
- Flag anomalous observations and explicit temperature-limit or rapid-rise conditions.
- Provide operator-facing alerts, risk factors, route comparison and what-if scenario outputs.
- Evaluate the analytical workflow on synthetic data and define the experiments required for real-world validation.

3.3 Scope and assumptions

The current release is a development/demo prototype. It does not implement production authentication and is not intended for safety-critical decisions. The supplied ZIP contains software modules and a simulator; it does not contain an ESP32 firmware sketch or a working physical DHT22 data-ingestion interface. The ESP32–DHT22 arrangement is therefore treated as the intended sensing configuration for the next implementation phase.

IV. PROPOSED SYSTEM ARCHITECTURE

The architecture is organized into sensing, data acquisition, analytics and application layers. The intended sensing layer uses an ESP32 microcontroller with a DHT22 sensor to measure temperature and relative humidity. A timestamped reading is associated

with a shipment and forwarded to the backend. In v0.1.0, the simulator supplies the readings and the API identifies the data mode as SIMULATED.

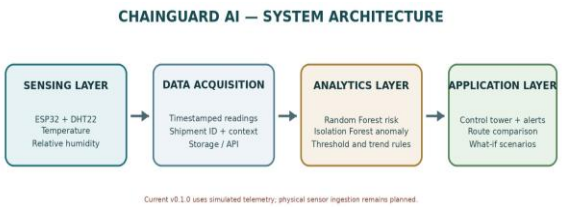


Fig. 2. Layered architecture showing the proposed sensor-to-application data path and the current simulation boundary.

4.1 Frontend and control tower

The frontend is built with React and TypeScript using Vite. It provides a logistics control tower, shipment views, monitoring visualisations, risk intelligence, anomaly centre, route intelligence, digital-twin scenarios, alerts and report export. Charting components support inspection of telemetry history and shipment status.

4.2 Backend and persistence

The backend uses FastAPI to expose REST endpoints for health status, dashboard data, shipment creation and retrieval, telemetry readings, simulator ticks, risk estimation, route comparison, digital-twin scenarios, alerts and CSV reporting. SQLAlchemy provides persistence; the project README describes SQLite as the local default and PostgreSQL as an optional deployment configuration.

4.3 Data flow

- A shipment is registered with product type, route details, vehicle identifier and configured temperature/humidity limits.
- A telemetry observation is received or generated by the simulator and stored with its timestamp and shipment identifier.
- The backend derives the model feature vector from the latest observation and shipment context.
- The Random Forest returns a positive-class probability, which is converted to a prototype score and category.
- The Isolation Forest and explicit threshold/trend rules generate anomaly and alert signals.
- The dashboard displays shipment state, risk category, heuristic factors, alerts and scenario outputs.

V. METHODOLOGY

5.1 Feature representation

The model uses six numeric inputs: current temperature T (°C), relative humidity H (%RH), transit duration τ (hours), delivery delay D (minutes), temperature exposure E and recent temperature change ΔT.

Transit duration is converted from minutes to hours. The backend computes a simplified exposure feature as $E = \max(0, T - T_{max}) \times \max(\tau, 0.1)$, where T_{max} is the configured upper temperature limit. This feature is a prototype approximation and not a product-specific thermal-damage or shelf-life model.

5.2 Random Forest risk estimation

The supplied ML engine creates 3,000 synthetic samples using NumPy with a fixed random seed. Feature values are generated from predefined distributions. A synthetic event probability is calculated from a hand-crafted logistic expression involving temperature, humidity, transit time, delay, exposure and positive temperature rise; binary labels are sampled from this probability. The data are split into training and validation subsets using a stratified 75:25 split.

A Random Forest Classifier is configured with 120 trees, maximum depth 9, minimum leaf size 4, balanced class weights and random_state=42. At inference, the model probability p is converted to a prototype score $R = 100p$. The code defines LOW for $R < 30$, MEDIUM for $30 \leq R < 60$, HIGH for $60 \leq R < 80$ and CRITICAL for $R \geq 80$. These are software display bands, not validated product-safety thresholds.

5.3 Isolation Forest and rule-based anomaly detection

An IsolationForest model is fitted on the synthetic training feature matrix with contamination set to 0.06. At inference, an observation is marked as an anomaly when the model predicts the outlier class. The backend also checks whether temperature is outside the shipment's configured range and whether the recent temperature rise exceeds a fixed threshold.

A statistical anomaly is an unusual feature combination, whereas a threshold alert indicates violation of a configured operating limit. Neither signal alone proves product damage.

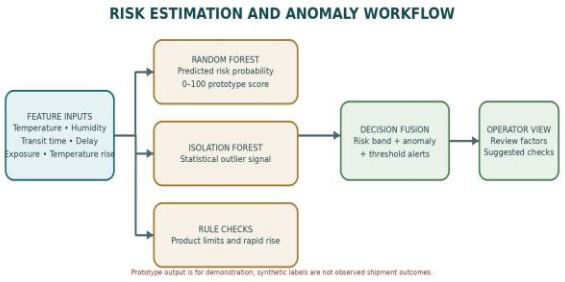


Fig. 3. Risk estimation, anomaly detection and rule-based alert workflow.

5.4 Explainability and recommended actions

The current implementation ranks heuristic impact values for temperature, humidity, transit duration, delay and temperature exposure, then presents the highest-ranked factors. These values are manually defined formulae; they are not SHAP values, causal effects or faithful decompositions of the Random Forest prediction. Recommended actions are rule-based messages, such as checking cooling equipment after a temperature-limit breach or reviewing route alternatives when delay is high.

5.5 Route comparison and scenario analysis

The route endpoint compares three illustrative candidate routes using fixed distance, travel time, cost, delay-risk and condition-risk inputs. A weighted composite cost is calculated and the minimum-cost candidate is recommended. The inputs and weights are simulated and heuristic; no live traffic or road-condition service is connected. The what-if endpoint evaluates scenarios such as an additional delay, a temperature excursion or an alternative-route assumption. It returns before/after risk estimates without saving the scenario to the live shipment. This is scenario simulation rather than a validated digital twin.

5.6 Mathematical formulation

Let $x = [T, H, \tau, D, E, \Delta T]$ denote the feature vector and $f_{RF}(x)$ the Random Forest positive-class probability. The prototype score is $R(x) = 100 \times f_{RF}(x)$. The simplified exposure term is $E = \max(0, T - T_{max}) \times \max(\tau, 0.1)$. For route candidate r, the prototype computes a weighted composite cost $C(r)$ from normalized travel time, cost, delay risk and condition risk, and selects $r^* = \arg\min_r C(r)$. The route weights and risk bands are configuration choices requiring calibration.

VI. SOFTWARE IMPLEMENTATION AND PROTOTYPE STATUS

Component	Implementation in v0.1.0	Status / interpretation
Frontend	React, TypeScript, Vite, Recharts	Dashboard and visualisation implemented
Backend API	Python FastAPI	REST endpoints implemented
Persistence	SQLAlchemy; SQLite default; PostgreSQL option	Persistence supported
Risk estimator	RandomForestClassifier (120 trees)	Trained on synthetic labels
Anomaly detector	IsolationForest (contamination 0.06)	Fitted on synthetic features
Alerts	Threshold and temperature-rise checks	Rule-based; simulator generates events
Route intelligence	Weighted heuristic over three routes	Illustrative inputs only
What-if module	Delay / excursion / route scenarios	Scenario simulation only
Physical sensing	ESP32 + DHT22 intended configuration	Firmware and live ingestion absent from supplied ZIP

The project README and manifest identify the current telemetry, route inputs and model-training labels as synthetic. The API health and dashboard responses also identify the data mode as SIMULATED. Therefore, the prototype demonstrates software integration and workflow behaviour, not a field-deployed cold-chain monitoring system.

VII. EXPERIMENTAL SETUP

Because the supplied software does not include real shipment measurements, the experiment in this section is explicitly a synthetic simulation. It was generated to demonstrate the model-training and evaluation workflow using the same six-feature schema and synthetic-label concept as the project. The experiment must not be interpreted as a field trial or as evidence of performance on real shipments.

7.1 Synthetic dataset generation

A reproducible dataset of 3,000 simulated observations was generated with a fixed random seed.

The features were temperature, relative humidity, transit duration, delivery delay, temperature exposure and recent temperature change. A synthetic event probability was generated using a logistic function of these features, and binary event labels were sampled from that probability. The overall positive-label prevalence in this generated dataset was 17.5%. A stratified 75:25 split produced 2,250 training observations and 750 held-out test observations.

7.2 Models and evaluation protocol

Two supervised models were compared: a Random Forest classifier configured with 120 trees, maximum depth 9 and minimum leaf size 4; and a logistic-regression baseline with standardized input features. Both models used class weighting to account for the imbalanced synthetic labels. The decision threshold was 0.50. Accuracy, precision, recall, F1-score, ROC-AUC and PR-AUC were calculated on the held-out synthetic partition. The test partition is independent of model fitting but not independent of the synthetic label-generation assumptions.

7.3 Synthetic benchmark results

Metric	Random Forest	Logistic Regression
Accuracy	85.60%	82.67%
Precision	59.83%	50.31%
Recall	53.44%	61.83%
F1-score	56.45%	55.48%
ROC-AUC	0.811	0.836
PR-AUC	0.620	0.663

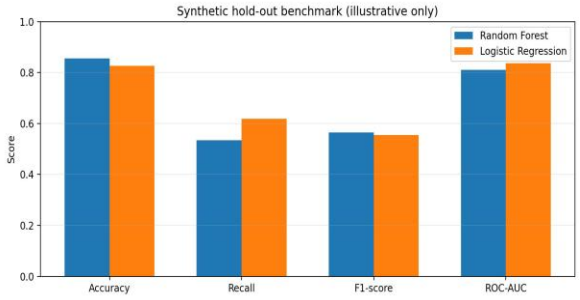


Fig. 4. Comparison of selected metrics on the held-out synthetic test partition; values are illustrative simulation results only.

In this generated experiment, Random Forest achieved higher accuracy and precision, whereas logistic regression achieved higher recall, ROC-AUC and PR-AUC. The difference illustrates that model selection depends on the operational objective: a system prioritising missed-event reduction may prefer a different threshold or model from one prioritising fewer false alert. The observed metrics are conditional on the synthetic feature distributions and label-generation function; they do not establish generalisation to actual cold-chain shipments.

7.4 Confusion matrix and interpretation

Random Forest (threshold 0.50)	Predicted non-event	Predicted event
Actual non-event	572	47
Actual event	61	70

The confusion matrix shows 70 true positives and 61 false negatives in the synthetic test set. This demonstrates why recall and false-negative analysis matter in risk-monitoring applications. In a real deployment, the cost of a missed excursion may be substantially different from the cost of an unnecessary inspection; the classification threshold should therefore be selected using domain-specific costs and validation data.

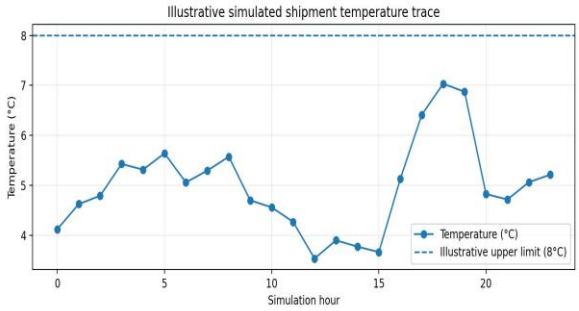


Fig. 5. Example simulated temperature trace with an illustrative upper-limit line. This is not a physical DHT22 recording.

VIII. RESULTS AND DISCUSSION

The synthetic benchmark confirms that the implemented feature schema can support a reproducible supervised-learning demonstration. It also shows a trade-off between precision and recall across the two compared models. The Random Forest achieved a higher F1-score by a small margin in this particular simulation, while logistic regression produced higher ROC-AUC and PR-AUC. No claim is made that one model is superior for actual cold-chain operations.

The implementation-level findings are functional: the software includes shipment management, simulated telemetry history, Random Forest risk scoring, Isolation Forest anomaly flags, threshold/trend alerts, heuristic route comparison, what-if scenario outputs and CSV reporting. These are software capabilities verified from the supplied project archive. Sensor accuracy, end-to-end latency, alert usefulness, model calibration and predictive performance on real shipments remain unmeasured.

The central limitation is the synthetic origin of both the current software labels and the experiment reported here. Synthetic data are useful for testing data pipelines, visualisations and model integration, but cannot establish whether the system predicts spoilage, food-safety events or pharmaceutical quality loss. A future study must use independently defined outcomes and shipment-grouped or time-aware data splits.

IX. LIMITATIONS, SAFETY AND FUTURE WORK

The current model is trained on synthetic data; its labels are not independently observed outcomes. The supplied software archive lacks physical ESP32 firmware and live DHT22 ingestion. Route comparison relies on fixed illustrative inputs. The factor-impact display is heuristic rather than a validated explanation method. The login component is a user-interface shell and is not production authentication. The prototype is not suitable for safety-critical decisions.

Future work will complete ESP32–DHT22 firmware and live ingestion, compare sensor readings against a calibrated reference instrument, collect representative shipment data, define independently verifiable event labels, compare suitable baselines and conduct shipment-grouped evaluation. Additional work should calibrate risk probabilities, use recognised explanation methods, connect live route information, harden security and conduct a controlled pilot.

Any deployment involving food, vaccines, pharmaceuticals or biological samples must use product-specific storage and transport limits established by manufacturers and applicable authorities. The system should support—not replace—qualified quality-management procedures. Secure communication, access control, audit trails, time synchronisation, calibration records, data retention and fault handling must be addressed before deployment.

X. CONCLUSION

This paper presented CHAINGUARD AI, a software-first framework for integrating shipment monitoring, risk estimation, anomaly detection, threshold alerts, route comparison and what-if scenario analysis. Source-code inspection confirms a React/TypeScript frontend, FastAPI backend, SQLAlchemy persistence, a Random Forest risk estimator and an Isolation Forest anomaly detector. A synthetic benchmark was added to demonstrate the evaluation workflow and to compare Random Forest with logistic regression. The experiment produced measurable but simulation-dependent results and must not be represented as real-world shipment validation.

The next research milestone is to connect the physical ESP32–DHT22 sensing setup, collect traceable sensor readings and independently labelled shipment outcomes, and repeat the evaluation using leakage-resistant splits and operationally relevant metrics. Only after this work can the predictive effectiveness and deployment value of CHAINGUARD AI be assessed.

REFERENCES

[1] Chaudhuri, I. Dukovska-Popovska, N. Subramanian, H. K. Chan, and R. Bai, "Decision-making in cold chain logistics using data analytics: A literature review," *The International Journal of Logistics Management*, vol. 29, no. 3, pp. 839–861, 2018, doi: 10.1108/IJLM-03-2017-0059.

[2] E. Badia-Melis, P. Mishra, and L. Ruiz-García, "Food traceability: new trends and recent advances. A review," *Food Control*, vol. 57, pp. 393–401, 2015. Bibliographic details should be verified against the publisher record before submission.

[3] W. A. Müller et al., "Enhancing Food Safety in the Cold Chain Through Internet of Things and Artificial Intelligence," *Journal of Food Science*, 2026, doi: 10.1111/1750-3841.70871. Verify author list, volume, issue and page/article details against the publisher record.

[4] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[5] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation Forest,” in *Proc. 8th IEEE International Conference on Data Mining*, 2008, pp. 413–422, doi: 10.1109/ICDM.2008.17.