DataGuardian

Decision Guard System for Safe Data-Driven Decisions

Project Documentation

Prepared by
Vishal Mankar

Year
2025

Description

DataGuardian is a system designed to prevent business decisions from being made on incomplete, biased, or misleading data.
It enforces rule-based validation, decision blocking, warnings with accountability, and full audit lineage before any decision is allowed.

DataGuardian – Project Documentation

---

1. Introduction

DataGuardian is a decision-guard system.

Its job is simple:

To make sure decisions are not made using bad, risky, or misleading data.

In many real-world projects, people trust dashboards and metrics without checking whether the data is:

- fresh
- complete
- large enough
- biased
- or even correct for the decision

This creates **false confidence**, which leads to **wrong decisions**.

DataGuardian solves this by checking the data **before** a decision is allowed.

---

2. Why This Problem Exists

2.1 Common Problems in Data-Driven Decisions

In companies and projects, these problems happen very often:

- Numbers look correct but data is old
- Results are based on very small sample size
- One customer or segment controls the result
- Vanity metrics look good but business is actually failing
- Wrong metric is used for the wrong decision

Example:

“Revenue increased by 12%”
But actually:

- Only 2 customers paid early
- Other customers are leaving
- Decision based on this metric becomes dangerous

Dashboards **do not stop this**.
They only show numbers.

---

3. What DataGuardian Solves

DataGuardian acts as a **gatekeeper** between data and decisions.

Instead of asking:

“What does the data say?”

It asks:

“Should we trust this data for this decision?”

It stops unsafe decisions **before they happen**.

---

4. Core Idea of DataGuardian

Every decision must pass **data trust checks**.

The system never assumes data is correct by default.
It is **skeptical by design**.

Only after validation does it allow a decision.

---

5. Decision Flow (How the System Works)

Step 1: Decision Request

The user provides:

- Decision type (pricing, marketing, growth, etc.)
- Metric to use
- Data context:
    - time range
    - sample size
    - data freshness
    - segmentation

This makes the decision **explicit**, not hidden.

Step 2: Guard Rule Evaluation

The system runs **rule-based checks** on the data.

Each rule checks a specific risk.

Rules do not guess or predict.
They **validate**.

Step 3: Decision Status

After evaluation, the system returns **one clear result**:

- **ALLOW** → data is safe
- **WARN** → data is risky
- **BLOCK** → data is unsafe

There are no soft suggestions.

Step 4: Override (Only for WARN)

- WARN decisions can be overridden
- User must give a reason
- Confidence is reduced
- Override is permanently logged

BLOCK decisions **cannot be overridden**.

---

6. Guard Rules Explained (Simple)

6.1 Data Freshness Rule

Checks if the data is too old.

Why:

- Old data can give wrong signals

Result:

- Too old → BLOCK or WARN

6.2 Partial Data Rule

Checks if data is incomplete.

Why:

- Incomplete data hides real trends

Result:

- Partial data → BLOCK

6.3 Sample Size Rule

Checks if there is enough data.

Why:

- Small samples create fake trends

Result:

- Too small → BLOCK

6.4 Comparison Validity Rule

Checks if two periods can be compared.

Why:

- Comparing unequal data creates false growth or loss

Result:

- Invalid comparison → WARN

6.5 Concentration (Bias) Rule

Checks if few entities dominate results.

Why:

- One customer can mislead overall performance

Result:

- High concentration → WARN

6.6 Segment Drift Rule

Checks if data composition changed.

Why:

- Change in audience can fake improvement or decline

Result:

- Large drift → WARN

6.7 Vanity Metric Rule

Checks if metric looks good but business is bad.

Why:

- Metrics like clicks or engagement can hide losses

Result:

- Vanity metric → WARN + counter-metric required

6.8 Metric–Decision Match Rule

Checks if metric fits the decision type.

Why:

- Wrong metric leads to wrong decision

Result:

- Mismatch → WARN or BLOCK

---

7. Confidence Score

Confidence score answers:

“How much can we trust this decision?”

Important:

- It is NOT a prediction
- It is NOT success probability
- It only measures **data trust**

Rules:

- Confidence only decreases
- Overrides always reduce confidence

- Lower confidence = higher risk

---

8. Override System (Accountability)

Overrides are serious actions.

To override:

- User must explain why
- Confidence is penalized
- Action is recorded forever

This ensures:

- No silent risk-taking
- Clear responsibility

---

9. Audit Lineage (Decision Memory)

Every decision creates **one immutable audit record**.

Audit record contains:

- Input data details
- Rule results
- Original status and confidence
- Final status after override
- Override reason (if any)
- Outcome status

This creates full traceability:

Data → Rules → Decision → Override → Outcome

Audit logs are append-only and cannot be edited.

---

10. Outcome Tracking

Outcomes can be marked later as:

- Positive
- Neutral
- Negative
- Unknown

This helps analyze:

- Which warnings were ignored
- Which overrides caused damage
- Decision quality over time

---

11. Design Principles

DataGuardian is built with strict principles:

- Correctness over speed
- Accountability over convenience
- Explainable logic over black boxes
- No machine learning
- No predictions
- No dashboard focus

---

12. What This Project Is Not

- Not an AI system
- Not a dashboard
- Not a forecasting tool
- Not a visualization project

It exists only to **protect decisions**.

---

13. Technology Used

- React
- TypeScript
- Tailwind CSS
- Rule-based logic
- Deployed on Vercel

---

14. Project Status

- Decision rules: ✅ Complete
- Override system: ✅ Complete
- Audit lineage: ✅ Complete

The core system is finished and functional.

---

15. Final Summary

DataGuardian helps prevent one of the most dangerous problems in data work:

Wrong decisions made with high confidence.

It forces discipline, records accountability, and makes risk visible.