Print this page to PDF (Ctrl/Cmd+P) Print to PDF [Full case study]

Sevendyne
Engineering firm · Since 2016
Technical one-pager

IoT sensor & temperature analytics

Engagement: Full-stack IoT platform **Stack:** Node.js · Spring Boot · Angular **Signal:** Temperature & device telemetry

Hardware gateways push readings into JVM calculation services and Node integration layers — Angular dashboards show trends, thresholds, and device health, not CSV exports.

Node.jsSpring BootAngularTelemetry pipelinesThreshold alerts
Pipeline

Device → gateway → serve → UI

Node.js accepts gateway payloads (batch + REST). Spring Boot runs roll-ups, threshold checks, and persistence. Angular polls REST aggregates and highlights alert conditions for operators.

Design decisions

Why split Node + Spring

Layer	Role	Why
Node.js edge	Fast ingest, protocol adapters	Iterate on messy device payloads quickly
Spring Boot core	Calculations, persistence, rules	Typed services for temperature math and audit
Angular UI	Operator dashboards	Reusable chart components per site/device

Reliability

IoT tactics

- Idempotent ingest keyed by `device_id + reading_ts`
- Gap detection when gateways miss heartbeats
- Alert rules versioned — threshold changes don’t rewrite history

Outcomes

Results

- End-to-end path from sensor hardware to operator dashboards
- Reusable pattern for JVM + JS IoT programmes
- Complements deeper C++/Laravel portfolios

IoT programme. Client identity available under NDA.

contact@sevendyne.com · sevendyne.com/case-studies