CAPABILITY STATEMENT

Software, automation, and data systems for dependable public services.

Lumo Systems helps public sector teams replace brittle manual processes with clear, maintainable digital services. We build software, automation, and data systems that make services easier to run, easier to govern, and easier to hand over.

EXECUTIVE SUMMARY

A specialist team delivering software, automation, and data systems where clarity, reliability, and auditability matter.

Lumo Systems delivers software, automation, and data systems for organisations that cannot afford messy operations. Our priorities are simple: make the service easier to run, make decisions easier to review, and leave the client with something they can genuinely own.

Software

Internal tools, portals, workflow applications, and service components designed for long-term maintainability rather than short-term patchwork.

Automation

Rules, checks, orchestration, and process automation that reduce manual handling while keeping controls visible and changeable.

Data Systems

Ingestion, validation, transformation, and structured storage for services that depend on accurate information moving between systems.

WHY LUMO

Direct engineering, fewer layers, and work you can actually take ownership of.

Many suppliers add layers between the client and the people doing the work. Lumo is set up differently. Clients work directly with experienced engineers from the start, which keeps communication clearer and delivery closer to the real operating need.

Direct delivery

You work directly with the people designing and building the service, not a chain of account management and handoffs.

No lock-in by design

We favour open standards, readable architecture, and full documentation so the service can be operated and extended without dependency.

Governance built in

We build in validation, traceability, and clear system behaviour from the start, so governance is part of the service rather than something added later.

Work that survives handover

We build with internal teams in mind, so support, transition, and future procurement are simpler rather than harder.

CORE CAPABILITY

What we build

We usually work where software, operations, and policy meet: the parts of a service that need to be usable for staff, clear to stakeholders, and dependable under day-to-day pressure.

Software systems

1. Operational tools for case handling, internal workflows, and service administration.
2. User-facing applications with clear interfaces and maintainable backend services.
3. Modular services designed to support safe updates, better change control, and long-term ownership.

Automation and data systems

1. Workflow automation that reduces manual reconciliation, repetitive handling, and avoidable delay.
2. Data processing pipelines that validate, transform, and route information between systems.
3. Rule-based services that make decisions more consistent, more reviewable, and easier to justify.

DELIVERY APPROACH

A practical delivery approach for services that need to work in the real world.

We usually start with a tightly scoped discovery or alpha, reduce uncertainty early, and then move into delivery with a clear view of ownership, controls, and operational support.

01

Discovery

Understand user needs, current process pain points, data movement, controls, and delivery risk.

02

Alpha

Test the riskiest assumptions, prove the shape of the service, and agree the delivery path.

03

Build

Develop the service with clear validation, documented rules, and maintainable architecture.

04

Handover

Provide documentation, structured transition support, and a service the organisation can own.

HOW WE WORK IN PRACTICE

A straightforward way of working for teams that need clarity and steady delivery.

We keep delivery simple: clear scope, direct communication, visible progress, and documentation alongside the build. That makes projects easier to manage and easier to hand over once the work is complete.

What clients can expect

1. Direct access to the engineers doing the work.
2. Defined scope, milestones, and outputs.
3. Regular check-ins and working progress updates.

How we reduce dependency

1. Readable systems and practical documentation.
2. Open standards where appropriate.
3. Structured handover to internal teams or future suppliers.

SELECTED EXPERIENCE

Examples of delivery in complex and regulated environments.

The following case studies reflect delivery led by our engineers in previous roles and projects, in environments where consistency, auditability, and operational reliability mattered.

01

Automated millions of daily decisions with a single, auditable data pipeline

Real-Time Data Processing & Automated Reconciliation

THE BRIEF

A client operating in a regulated, high-frequency environment needed a system that could process live operational data at scale, apply decision logic in real time, and maintain an accurate historical record for audit and analysis.

Existing approaches relied on fragmented processing and manual reconciliation, which introduced delay, inconsistency, and limited traceability.

WHAT WE DELIVERED

- A unified processing model for both live and historical data.
- Deterministic decision logic so replay and production behaved the same way.
- Structured logging and storage for end-to-end traceability.
- Automated reconciliation to remove manual validation steps.

Outcome

The client gained a stable processing layer capable of handling millions of events per day, with consistent outputs across live and historical data and a complete audit trail for review.

Why it matters

This kind of work translates directly to public service environments where accuracy, traceability, and confidence in system behaviour are non-negotiable.

02

Standardised eligibility decisions with a rule-based service and clear audit trails

Eligibility Assessment & Rule-Based Decision System

THE BRIEF

The client needed a structured tool to govern eligibility decisions across a high volume of cases. Manual processes had led to variation in outcomes, weak validation, and limited visibility during audit and review.

The service needed to align decisions with defined policy, reduce misuse and invalid outcomes, and provide clear justification for each decision made.

WHAT WE DELIVERED

- A rule-based decision engine enforcing consistent eligibility logic.
- Structured validation layers for input quality and control.
- Clear mapping between inputs, rules, and outcomes.
- Comprehensive logging to support audit and review.

Outcome

The result was a more controlled and scalable process, with fewer invalid outcomes, faster policy updates, and stronger confidence in decision consistency across teams.

Why it matters

Public services often depend on decisions being applied fairly and consistently. This kind of system helps teams make that consistency visible rather than assumed.

03

Built a modular service platform that made change easier and kept the client in control

Modular Service Architecture & Workflow Integration

THE BRIEF

The client needed a public-facing, data-driven service that pulled information from multiple external sources and presented it through a clear, accessible interface.

As requirements evolved, tightly coupled components made even small updates expensive and disruptive. The platform needed a cleaner structure that would support change over time without locking the client into opaque systems.

WHAT WE DELIVERED

- A modular web application with clear separation between frontend, backend, and integration layers.
- Validation and normalisation for inconsistent incoming data.
- Structured interfaces for adding new features and data sources safely.
- Transparent system behaviour and documentation to support ownership.

Outcome

The service became easier to extend, cheaper to update, and less risky to change. The client kept full ownership of the code, the data, and the way the service behaved.

Why it matters

For public sector teams, this means change can happen without a rebuild every time policy, content, or user need moves.

ASSURANCE & HANDOVER

Built to be understood, run, and handed over properly.

We design for continuity after delivery. That means readable architecture, practical documentation, transparent rules, and a handover process that leaves the organisation in control.

Typical assurance principles

1. Clear validation and review points for inputs, rules, and outputs.
2. Traceable workflows and event logging where decisions or state changes matter.
3. Structured documentation to support internal support teams, auditors, and future suppliers.

Typical handover outputs

1. Technical documentation, architecture notes, and deployment guidance.
2. Codebase transfer, configuration notes, and operational runbooks where needed.
3. Transition support for internal teams or onward delivery partners.

Lumo Systems

Capability Statement

Procurement Contact

procurement@lumosystems.co.uk
Lumo Systems Ltd
Registered in England & Wales
Company No. 12224860