Reimagining Rules Management for a leading financial services company: **From Hardcoded Logic to a Configurable, Runtime-Driven Rules Engine**

The Problem

The organizations core business logic had grown into a tangle of conditional rules hardcoded directly in application code. Every product variation, eligibility check, and customer-specific exception lived inside if/else chains scattered across the codebase, with no single place to view, trace, or reason about how a decision was actually made. Understanding or changing a single rule meant hunting through multiple services, and even small policy or product changes required a full development, testing, and release cycle. As the rule set grew more complex, the impact compounded: the team estimates 60–70% of productive development time was being consumed by debugging and root-cause analysis rather than building new capability, and extending the system for new customers or products became slower and riskier with each release.

Inadev designed and built a highly customizable, configurable Rules Engine that extracts conditional logic out of hardcoded application code and into a centralized, runtime-configurable layer. Rules that once lived inside source code are now modeled, versioned, and managed independently of the application, so business and technical teams can view exactly how a decision is derived, change a rule without a deployment, and extend the logic for new customers without touching core code. The result is a rules layer built for transparency, speed of change, and long-term extensibility.



Solutions Highlights

- **Externalized Conditional Logic:** Extracted scattered, hardcoded conditional flows out of application code into a single, structured rules layer — replacing tribal knowledge and buried if/else chains with a definition of business logic that both engineers and business stakeholders can inspect.
- **Runtime-Configurable Rules:** Enabled rules to be created, modified, and versioned at runtime, removing the code-build-deploy cycle from routine rule changes and letting teams respond to business and product changes far faster.
- **Reduced Debugging & Analysis Overhead:** Centralized, traceable rule execution replaced logic spread across the codebase, cutting the time engineers spend tracing conditional paths to isolate and resolve issues.
- **Built for Extensibility:** Structured the engine so new conditions, products, and customer-specific variations can be configured rather than custom-coded, shortening the path to onboarding new Financial customers and products.
- **Improved Governance & Maintainability:** Gave the business a clear, auditable view of active rules and their history, reducing reliance on developers to explain or verify how a given decision was reached.



Business Outcomes

20–25%

of dev time reclaimed

productive engineering time previously lost to tracing hardcoded conditional logic is now available for feature delivery.

50%

faster rule changes

business rules that once required a code change, build, and release cycle can now be updated and deployed at runtime.

40%

faster issue debugging & analysis

centralized, traceable rule execution replaces scattered conditional code, cutting time to isolate and resolve production issues.

10+

configurable rule extensions

new customer-specific variations can be configured rather than custom-coded, shortening onboarding for new Boost Financial customers.