Inter IIT
Tech Meet 14.0

LAMINAR

Streamlining workflows

.pathway

Team 54

Contents

1 Introduction

The fintech ecosystem has been rapidly evolving over the last few years. According to EY, there are approximately **10,000 fintechs in India** [1]. However, this growth has also introduced the burden of operationally heavy workflows that divert attention from core functionalities. Our research indicates that many early and mid-stage fintech startups spend **considerable time, effort and capital on peripheral processes**, such as Service Level Agreement (SLA) Monitoring, Reconciliation, Market Research, Compliance, and Regulatory and Operational Reporting. While automation tools exist, they are fragmented, not fully autonomous, and often depend on slow batch-processing systems. Furthermore, Root Cause Analysis (RCA) is frequently performed manually and with delays.

We found that these operational gaps have a direct impact on a startup's ability to scale efficiently. At the same time, workflows vary widely across fintech segments, underscoring the need for an adaptable platform. This presents a substantial market opportunity, one that led us to build Laminar.

Laminar is a low-code operational intelligence platform designed for fintech startups. It can automatically ingest live data, parse documents, predict SLA risks, perform **streaming-based** reconciliation and utilize **agentic reasoning** to provide **proactive Root Cause Analysis**. It is built on top of **Pathway**, a high-performance streaming computation engine enabling Laminar to process billions of events with **deterministic** outputs, ensuring that workflows behave consistently under scale. Laminar aims to bridge market gaps by transforming traditionally manual, slow and reactive peripheral fintech workflows into **real-time, automated and intelligent systems.**

2 Uniqueness

Easy to use

With Laminar, users can ingest legal documents and translate complex clauses into actionable business metrics.Powered by **Pathway's 300+ data connectors**, Laminar brings essential SRE processes into one **unified workspace**, letting teams build context-aware workflows through an intuitive **drag-and-drop canvas**. Unlike fragmented SRE tools with steep learning curves, Laminar's intelligent helper assistant guides workflow creation with **auto suggestions** and **accept/reject** controls, making sophisticated SRE automation almost effortless.

Preemptive Alerts

With **Pathway's streaming architecture**, our models learn from every new event as it happens, refining their understanding of system behavior and financial impact. This continuous learning lets us forecast issues early and send preemptive alerts, giving teams time to respond before customers are affected; far beyond the reactive, post-incident alerts offered by traditional SRE tools.

Root Cause Analysis

Leveraging the extensive power of **agentic AI** and Pathway's compatibility with **streaming data sources**, our platform gets rid of the need to manually correlate outputs from various SRE tools when an incident occurs, through our Root Cause Analysis agent, which analyzes relevant logs and historical issues to diagnose the root cause. It then suggests confidence-scored fixes and either executes them based on permission or invokes **human in the loop.**

Pratyush Rai
CEO, Merlin AI

> What stands out is Laminar's real-time AI readiness, its agents don't just react, they adapt on the fly. That's the future of intelligent financial operations

3 Problem Understanding & Use Cases

3.1 Service Level Agreements (SLA)

Service Level Agreements (SLA) (and the operationally related SLO) govern interaction between service providers and their clients. They define **measurable expectations** such as uptime, latency thresholds, error rates, response and resolution time, ensuring that both parties have a clear understanding of service quality and responsibilities. In fintech domains such as payments, lending, account aggregation and identity/KYC APIs, SLAs are **business-critical**: downtime or high latency immediately blocks transactions, causes failed authorisations and translates directly into **revenue loss**, **customer complaints** and **reputational damage**. SLA breaches expose the service provider to **legal actions**, adversely affect **customer trust** and **reliability**.

3.1.1 Validation Within the Account Aggregator Ecosystem

Account Aggregators help customers securely share their financial data, with consent, between banks and lenders. They make financial data transfer faster, safer and standardised.

Tejinder Singh
CEO, CAMSfinserv

> With low AA margins, Datadog and Splunk are prohibitively expensive, and we still need to spot an SLA breach before the customer feels it and communicate it clearly.

Vamsi Madhav
CEO, Finvu AA

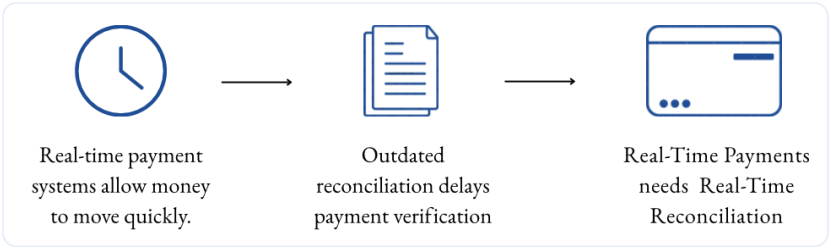
> Partners need failures explained in business impact, not just tech jargon, which means delivering fast, complete RCAs with clear causes, consequences, and corrective steps.

Figure 1: Pain points of real industry highlighted by CEOs of prominent Account Aggregators

Laminar solves the aforementioned pain points of real industry with ease by creating workflows that harness **Pathway's efficient live indexing** for ingesting logs, autonomous RCA using **agentic AI** and providing business level translation of technical metrics using **intelligent financial agents**.

3.2 Financial Data Reconciliation

Reconciliation is the process of cross-checking two or more sets of records, primarily a company's internal financial records against external sources, to identify and resolve any discrepancies, mandate the maintenance of accurate and complete financial records to ensure transparency and financial integrity [2] [3] [4].



20m transactions daily by companies like Revolut. At this scale, small error, big risk.

$85m discrepency between money held and money owed, Synapse had to file bankruptcy.

49% of world's real time payments happen in India.

In the context of real-time payment (RTP) systems, reconciliation becomes critical. RTPs operate at high velocity, with transactions settling in seconds, which demands equally **fast verification** and **error detection**. However, many financial institutions still rely on manual or outdated reconciliation methods that cannot keep pace with the speed and complexity of RTP flows. This gap leads to increased operational risk, delayed fraud detection, challenges in liquidity management and ultimately reduced customer trust. As McKinsey highlights, banks and fintech companies will need to adopt next-generation infrastructure to remain competitive and manage the growing operational burden of real-time payments [5]. While automated reconciliation tools such as Xero and QuickBooks help identify discrepancies and correct minor errors, they are fundamentally **not real-time systems**. Teams report spending nearly 40% of their time processing transactions due to slow, **batch-based workflows** [6]. Even with automation, 48% of finance teams say **fragmented data** remains their biggest obstacle to closing their books [7]. These limitations show that existing tools fall short in meeting the continuous, high-throughput demands of modern RTP environments.

Rhythm Pathak
Senior Consultant, Osfin AI

> We handle reconciliation in-house, and we see firsthand how complex and high-pressure it becomes in real-time environments, this is a problem that truly needs attention.

3.3 Sandbox (Future Scope)

Globally, all regulatory bodies demand compliance for the prevention of frauds, leakages, have defined frame work to mitigate potential threats. Similarly, in India the RBI Regulatory Sandbox is a controlled testing environment where fintechs can safely trial new or innovative financial products under RBI's supervision before launching them to the public [8]. Most fintechs need a strong internal testing and compliance engine to ensure there are no compliance threats. They typically do this by hiring a **Chief Compliance Officer** or by using compliance management systems like Regtrack and Software Mill. [9] [1] [10].

A developer's sandbox is a **testing environment** that utilizes **Laminar's intelligent alert** and **root cause analysis (RCA) agent** to monitor compliance and platform failures in real-time. The platform provides nodes to **generate dummy data** for testing. You can select the type of fintech you are working with and the regulatory rules you need to comply with, allowing for effective testing. If any failures occur during testing, Laminar alerts you in real time, provides reasons for the issues, so you can address them quickly.

3.3.1 Regulatory Barrier

India's fintech ecosystem is marked by **diverse business models** spanning payments, credit, wealth management, insurance, digital assets, intermediary/broking/advisory services, cryptocurrencies and more. Each of these models are subject to different regulatory regimes depending on the nature of the activity and the financial product or service involved. The principal regulators of fintech in India are: RBI , SEBI , IRDAI , PFRDA and FIU-IND. Apart from the above-mentioned key regulators there are few others [11].
Therefore, building a compliance sandbox requires **significant regulatory mapping** and **industry-level standardization**, which Laminar plans to implement in later phases.

4 Solution Overview

4.1 Creating Workflow

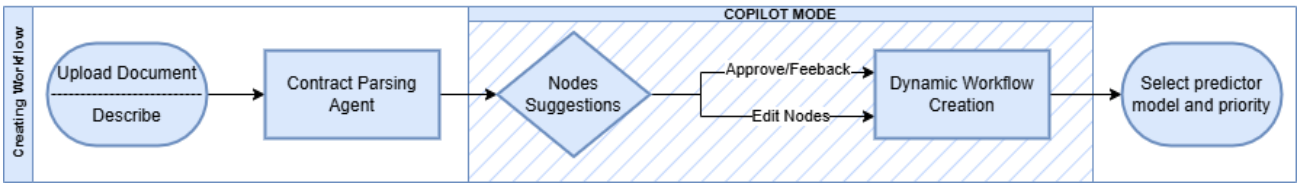


Figure 2: Creating Workflow with the User in CoPilot Mode

4.1.1 Low-code Canvas

Laminar provides a **drag-and-drop** visual interface where users can build data pipelines by connecting **pre-configured nodes**. 12.3.1 Pathway compiles this visual node-based interface and converts it into a production-grade data processing pipeline without any manual code optimization needed.

4.1.2 Contract Parser Agent (Low Code Pipeline Generation)

This agent transforms unstructured documents (like SLAs) that are uploaded to Laminar into important performance metrics. It then generates the complete monitoring workflow with the user in **copilot mode**. The agent can also take description of metrics from user without uploading any document , the agent will communicate accordingly to find the best pipeline for the metrics.

- **Phase 0**: After upload, documents are parsed and important **trackable performance metrics** are extracted.
- **Phase 1**: The agent interacts with the user to determine the appropriate data input format.
- **Phase 2**: The agent infers the **macro-level logic** and **iteratively** constructs the complete workflow, node by node, while allowing the user to provide feedback throughout the process for full control.

4.2 Stream Forecasting

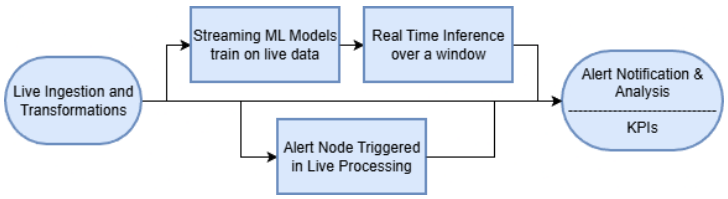


Figure 3: Data Flow and Alert Notifications

Using Pathway connectors, the Laminar handles **live ingestion** from various sources and then performs initial stateful transformations. We provide a **live**, node-level view of flowing tables and transformations, enabling **instant visibility** into pipeline health as data moves through the system. Data then flows into **Streaming ML Models** 12.4. We currently provide three ML models, namely:

1. **Time-series Dense Encoder (TiDE)** - suited for speed and low latency for long forecasting[12].
2. **Mamba** - suited for high accuracy, with the trade-off of being computationally heavy.[13]
3. **Adaptive Random Forect (ARF)** - provides a middle-ground balance between speed and accuracy.[14]

The ML models predict the value of a metric at time $t + \Delta T$. The predicted metrics are passed through a **filtering layer**, where any metric that violates a predefined threshold generates an alert, which is then displayed on the dashboard. The real-time data also flows through the standard workflow pipeline and if a hard threshold is crossed immediately, an **Alert Node** is triggered, capturing violations instantly. The alert will also initiate **Root Cause Analysis.**

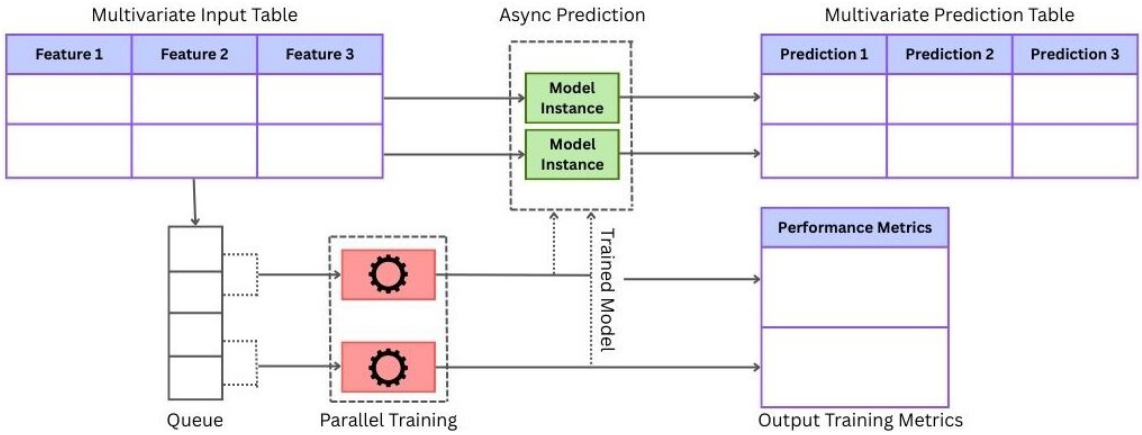


Figure 4: Streaming ML Architecture

4.3 RCA & Runbook

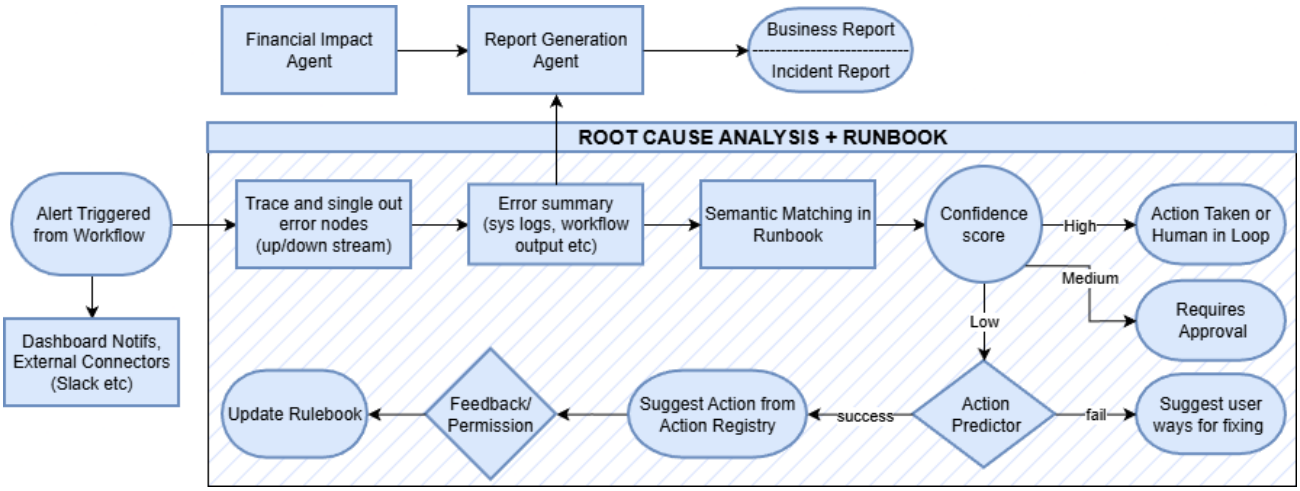


Figure 5: RCA and Runbook

4.3.1 RCA Agent

The RCA agent autonomously investigates incidents, when triggered by either **predicted breaches** (from the Streaming ML models) or **actual violations**. It queries OpenTelemetry logs and identifies slow services, checks recent deployments or configuration changes, along with both upstream and downstream dependencies. It isolates the root cause (whether it be a faulty internal service node or an external API failing) and sends an alert to the user. 12.4.2

4.3.2 Runbook

A key feature of the RCA agent is querying the Runbook for similar issues that have been encountered in the past. It performs **semantic search using pathway** over the Runbook and finds known fixes if the issue is a repeat. It forwards this output to the report generation agent along with a **confidence score** depending on how good a match it found.

4.3.3 Financial Impact Agent, and Report Generation Agent

This agent correlates transaction logs, revenue reference data, SLA penalty clauses, etc. to understand the lost revenue from failed, mismanaged transactions or potential SLA breaches based on their duration. This calculation updates in real-time as **incidents and outages progress**, enabling live financial tracking on the dashboard and in the reports. After RCA completion, the report generation agent uses a **LangGraph workflow** based on multiple **sub-agents**, to analyze the incident context and the RCA agent's output, to create a structured report.

5 System Architecture

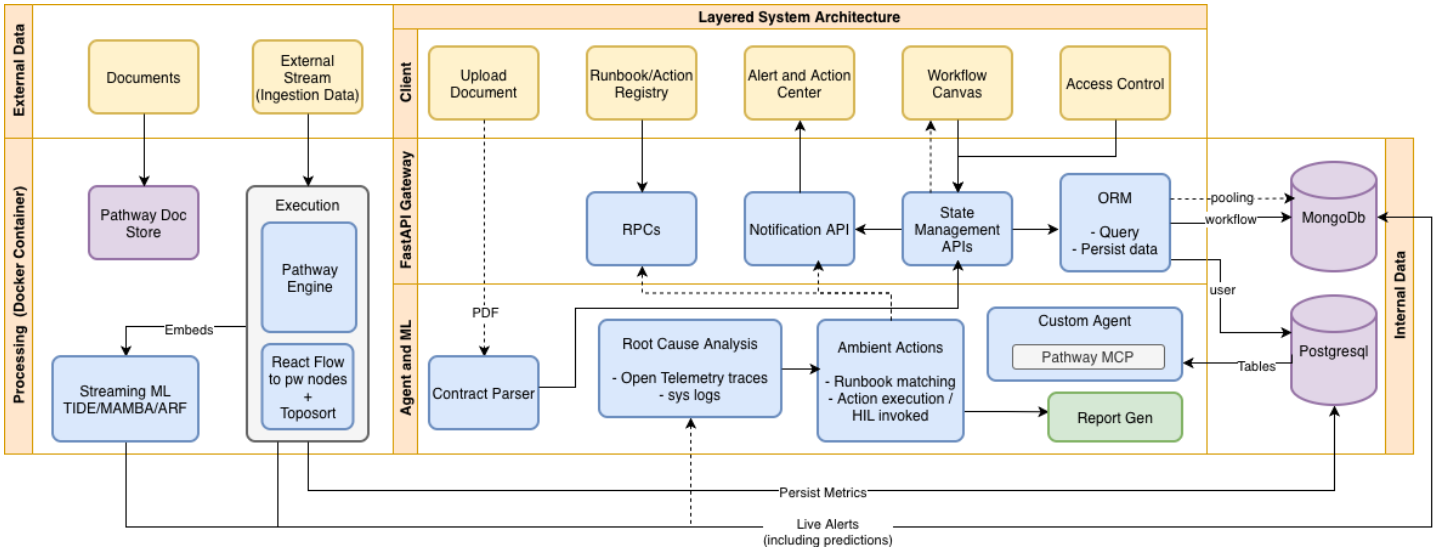


Figure 6: System Architecture (Appendix: 12.5)

6 Implementation Details and Production Bar

6.1 Streaming Ingestion and Live Indexing

Laminar takes advantage of **Pathway's extensive connector ecosystem**, of various input connectors including Kafka, Kinesis, PostgreSQL, MongoDB and the like, along with corresponding output connectors. This enables us to integrate multiple streams of data without the need for any custom adapter code. The streaming architecture constantly monitors directories for document ingestion, and Pathway's schema validation ensures type safety across all connector types, **preventing any runtime failures** from mismatched data formats, all the while **maintaining sub-second ingestion latency**.

6.2 Workflow and Agent Resiliance

Workflow Resilience

Unfinished workflows can be saved as drafts in the front-end. This is done through React state management, automatically persisting incomplete node configurations during workflow construction. During production, check-pointing of the pipeline is implemented. There are versions of the pipelines that are auto-saved and it is possible to revert back to a version anytime.

Agent Resilience

Agents implement a three-tier fallback system for reliability. When incidents occur, the agent performs **Rulebook matching**, trying to semantically find historical solutions to the problem and automatically executes fixes of the *low-risk, high-confidence* nature. For partial matches, the system alerts users about suggested remediations with confidence scores. New issues with no historical precedent trigger critical alerts and the rulebook is updated automatically after the user executes the fix. Credential exposure is prevented by using HMAC-signed **RPC Calls**, with secrets safeguarded through environment variable injection and Docker secrets management.

6.3 Overview of Guardrails

- **Layer 1 – Network & Protocol:** Makes sure the system only connects to safe and approved websites. It protects the connection and checks each URL before allowing access.
- **Layer 2 – Access Control:** Ensures that people can only do what their role allows. It keeps users within their data limits and applies strict identity checks so no one reaches information they shouldn't.
- **Layer 3 – Data & Content Analysis:** Scans messages using advanced language tools to detect harmful prompts or attempts to trick the system. Sensitive information is instantly hidden or blocked if it appears.
- **Layer 4 – Execution & Process:** Keeps actions safe by requiring a human review for anything risky. Session credentials are often rotated to prevent anyone from taking over a session.

Please refer to appendix 12.6 for more technical details about guardrails.

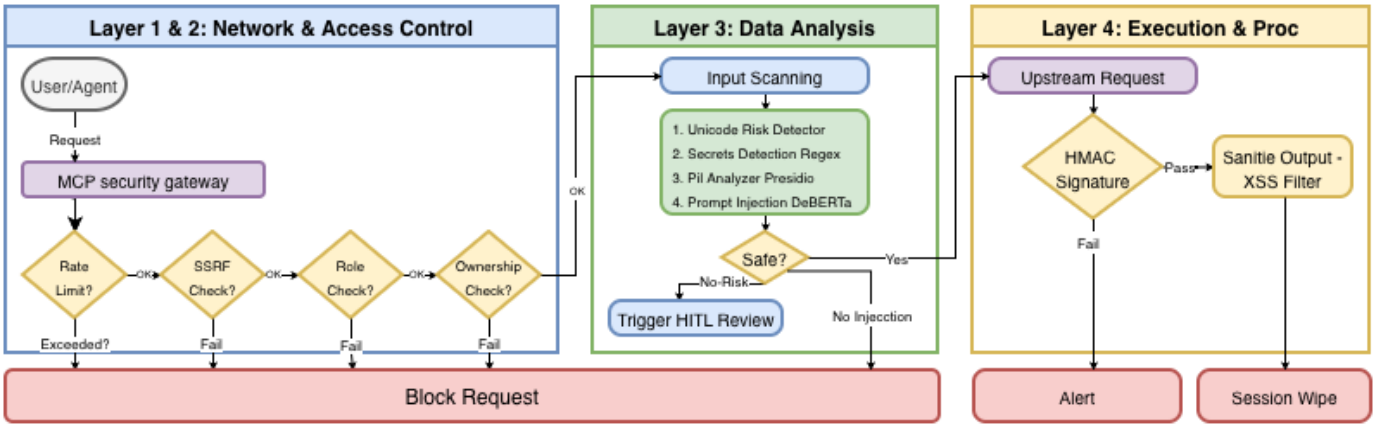


Figure 7: Multi-Layer Security Gateway Architecture

6.4 Deployment Plan

In production, the deployment architecture will run on the **Google Cloud Platform (GCP)** using a **Virtual Machine** setup. The system follows a **microservices-based** design, where each functional component is packaged as an independent container. Services run in their own isolated Docker containers, including a Pathway runtime and a lightweight SQL database for managing local state. This modular structure ensures loose coupling, independent deployment, and fault isolation. For each pipeline we spin up a new container, enabling seamless horizontal scaling.

7 Results and Metrics

7.1 Streaming ML Performance Metrics

Model	Avg MAE	Avg Latency (ms)	Avg RAM (MB)
ARF	2.457e-3	1.4663	1533.3681
TiDE	3.522e-3	0.4945	1549.1189
Mamba	1.248e-3	3.8764	1548.4453

Table 1: Performance Metrics Summary
Latency over the simulation period of 60 minutes

7.2 LLM Agent Performance Metrics

Feature	LLM Used	Time (s)	Cost (USD)
Contract Parser	`Claude-sonnet-4.5`	–*	0.1 – 0.15
RCA	`o1`	8 – 10	0.1 – 0.2
Runbook**	`gemini-2.5-flash`	12	0.02

Table 2: LLMs Metrics
(*interactive feature) (**Action registry)

8 Obstacles and Learnings

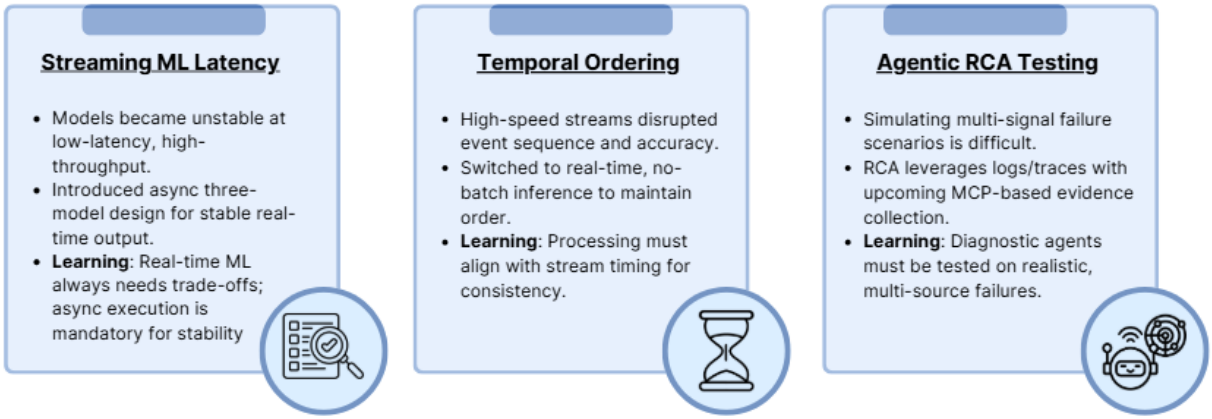


Figure 8: Solutions and Learnings

9 User Interface

The application provides a clean, intuitive experience with a **drag-and-drop** workflow builder using **React Flow**, enabling users to visually assemble pipelines and understand data transformations in real time through hover-based previews. A live dashboard presents **continuous KPIs, alerts, and breach notifications**, ensuring the **human-in-the-loop** stays fully informed. The workflow page includes version history, logs, and execution traces, improving transparency and auditability of every change. Security and governance are enforced using **RBAC**, while accessibility follows **WCAG AAA** guidelines to support all users. The Runbook feature securely manages secret keys, enabling automated actions without compromising safety. A detailed overview of the User Interface has been mentioned in the Appendix 12.7 .

10 Conclusion

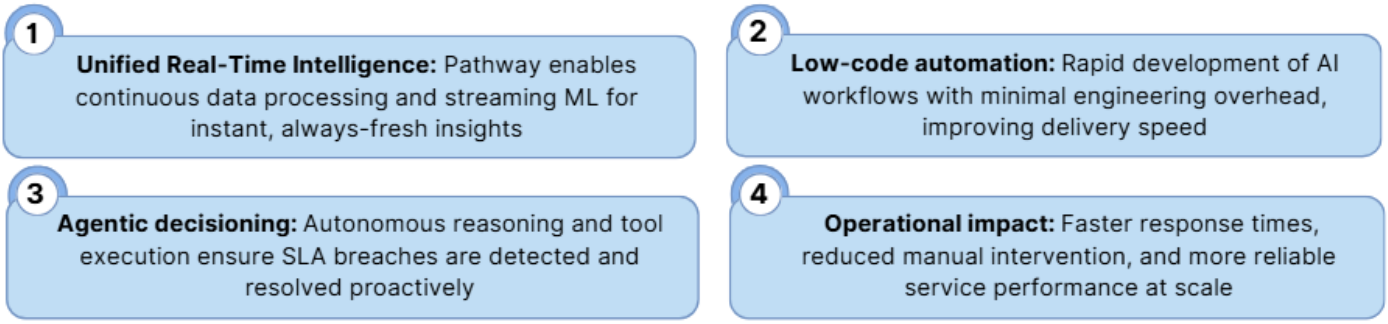


Figure 9: Conclusions

11 Future Work

Baby Dragon Hatchling Integration

Unlike standard LLMs constrained by fixed token buffers, BDH stores memory in **dynamic synaptic weights**, enabling our Root Cause Analysis (RCA) agents to correlate disparate log events over indefinitely long horizons, crucial for detecting complex, "slow-bleed" failure patterns like memory leaks that span hours or days. Furthermore, BDH directly addresses the stringent **explainability** requirements of the fintech sector through its inherent monosemanticity, where specific neurons activate for distinct concepts (e.g., 'DatabaseLatency'), offering a "white-box" alternative to opaque models and ensuring compliance with regulators like RBI and SEBI.

Crucially, the viability of this architecture is reinforced by recent independent experiments from the open-source community. An optimized E-BDH[15] configuration for computer vision demonstrated a **35x** training speedup over early baselines, and in CIFAR-10 tests a 5.5-point accuracy advantage with **44% fewer parameters** confirms that the BDH family is not only theoretically sound but empirically superior in constrained environments.

Laminar can pair this efficiency with BDH's fast-weight adaptability, allowing models to learn **client-specific traffic** topologies in real time without expensive retraining. The result is a highly efficient, sparse-activation, fully explainable system deployable directly within client VPCs, bridging the gap between high-performance automated remediation and the deep "trust-and-understand" requirements of financial infrastructure.

12 Appendix

12.1 References

[1] *The Role of FinTech in Building Viksit Bharat*. EY India. 2024. URL: https://www.ey.com/content/dam/ey-unified-site/ey-com/en-in/insights/financial-services/documents/ey-the-role-of-fintech-in-building-viksit-bharat.pdf.

[2] *Revolut's revenues surpass $2.2 bn, with record profits of $545 m in 2023*. Accessed: 2025-12-07. Revolut. 2024. URL: https://www.revolut.com/en-HR/news/revolut_s_revenues_surpass_2_2bn_with_record_profits_of_545m_in_2023/.

[3] *Synapse Faces CFPB Over Fund Reconciliation Gaps*. Accessed: 2025-12-07. PYMNTS. 2025. URL: https://www.pymnts.com/news/cfpb/2025/synapse-faces-cfpb-over-fund-reconciliation-gaps/.

[4] *Real-Time Payments in India*. Accessed: 2025-12-07. ACI Worldwide. 2024. URL: https://www.aciworldwide.com/real-time/india.

[5] *Global payments in 2024: Simpler interfaces, complex reality*. Accessed: 2025-12-07. McKinsey Company. 2024. URL: https://www.mckinsey.com/industries/financial-services/our-insights/global-payments-in-2024-simpler-interfaces-complex-reality.

[6] *Metric of the Month: Time Allocation in Finance*. CFO.com. 2024. URL: https://www.cfo.com/news/metric-of-the-month-time-allocation-in-finance/655989/.

[7] *The 2022 SaaS CFO Guide to FinOps Efficiency*. Accessed: 2025-12-07. Sage Intacct. 2022. URL: https://www.sage.com/en-us/sage-business-cloud/intacct/resources/ebooks/2022-saas-cfo-guide-to-finops-efficiency/.

[8] *FAQs on Payment Systems in India*. Reserve Bank of India. 2024. URL: https://www.rbi.org.in/commonman/English/scripts/FAQs.aspx?Id=3822.

[9] *NEW STUDY: 73% of FinTech Startups Fail Due to Regulatory Challenges*. Accessed: 2025-12-07. PR Newswire. 2025. URL: https://www.prnewswire.com/news-releases/new-study-73-of-fintech-startups-fail-due-to-regulatory-challenges-302421486.html.

[10] Nandan Sheth. *Fintech's Next Big Challenge? Thriving In An Era Of Regulatory Uncertainty*. Accessed: 2025-12-07. Forbes. 2025. URL: https://www.forbes.com/sites/nandansheth/2025/03/05/fintechs-next-big-challenge-thriving-in-an-era-of-regulatory-uncertainty/.

[11] *FinTech in India: An Overview of the Current Regulatory Landscape*. Argus Partners. 2024. URL: https://www.argus-p.com/papers-publications/thought-paper/fintech-in-india-an-overview-of-the-current-regulatory-landscape/.

[12] Abhimanyu Das et al. "Long-term Forecasting with TiDE: Time-series Dense Encoder". In: *arXiv preprint* (2023). arXiv:2304.08424. eprint: arXiv:2304.08424.

[13] Haoyu Ma et al. "A Mamba Foundation Model for Time Series Forecasting". In: *arXiv preprint* (2024). arXiv:2411.02941. eprint: arXiv:2411.02941.

[14] Heitor M. Gomes et al. "Adaptive Random Forests for Evolving Data Stream Classification". In: *Machine Learning* 106.6 (2017), pp. 1469–1495. DOI: 10.1007/s10994-017-5642-8.

[15] takzen. *vision-bdh: An experimental research framework in PyTorch for adapting the Baby Dragon Hatchling architecture to computer vision tasks*. Accessed: 2025-12-07. 2025. URL: https://github.com/takzen/vision-bdh.

12.2 List of Abbreviations and Acronyms

Short	Full Form	Short	Full Form	Short	Full Form
AA	Account Aggregator	FIP	Financial Information Provider	FIU	Financial Information User
HIL / HITL	Human-in-the-Loop	HMAC	Hash-Based Message Authentication Code	IRDAI	Insurance Regulatory and Development Authority of India
KPI	Key Performance Indicator	NBFC	Non-Banking Financial Company	ORM	Object-Relational Mapping
OTEL	OpenTelemetry	PFRDA	Pension Fund Regulatory and Development Authority	PII	Personally Identifiable Information
PVC	Persistent Volume Claim	RPC	Remote Procedure Call	RTP	Real-Time Payments
SLO	Service Level Objective	SRO	Site Reliability Engineering	SSRF	Server-Side Request Forgery

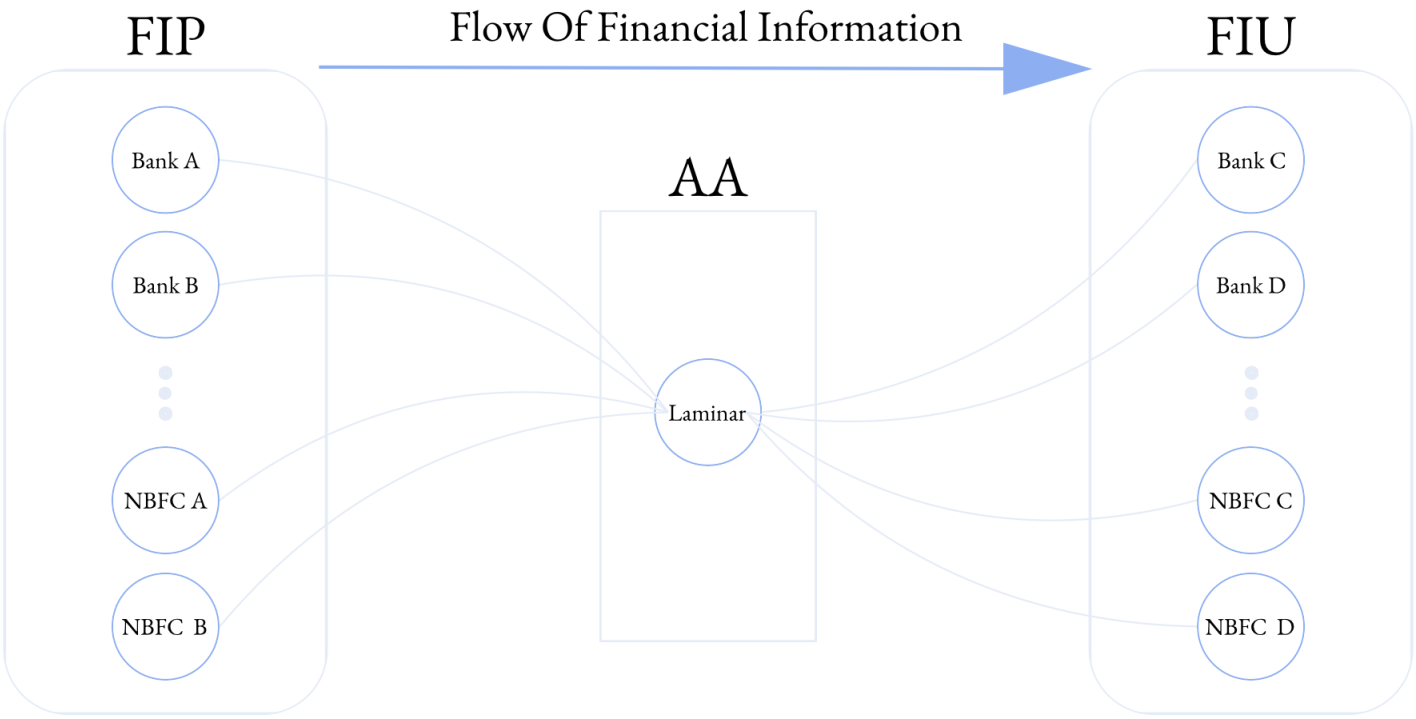


Figure 10: Flow of consent-based financial information from Financial Information Provider (FIP) to Financial Information User (FIU) through Account Aggregator (AA)

12.3 Platform Design

12.3.1 Low-Code Canvas

Laminar provides a node-based, drag-and-drop interface that lets you build complex data workflows by snapping together small, declarative elements. Under the hood, this is a JSON node that encodes its behaviour, dependency relationships, and parameterization. During run, the pipeline execution service serializes the configuration, and constructs a directed acyclic graph (DAG), through topological sorting of node dependencies. Each node function is mapped to a corresponding Pathway transformation function, forming a sequence of table operations that Pathway's engine executes. Thus, Pathway forms the computational backbone of Laminar, making it a production-grade data processing pipeline.

12.3.2 Core Node Categories

Laminar offers 40+ connector nodes for diverse data sources. These include data streams (eg: Kafka, Kinesis, etc.), databases (PostgreSQL, MongoDB, etc.), cloud storage (S3, Azure) and a number of external API connectors. Pathway's streaming architecture enables real-time data ingestion and exactly-once processing.

Custom Node Development: Laminar supports extensibility through a modular node system that allows developers to introduce new node types while remaining fully compatible with Pathway's streaming engine. A custom node is defined through a lightweight schema, mapped to a Pathway function, and incorporated automatically into the pipeline graph.
This architecture provides complete flexibility to developers, allowing them to introduce domain-specific capabilities, such as proprietary data sources or advanced analytics, all while maintaining Laminar's unified streaming execution model.

Category	Purpose	Representative Capabilities
Table Operations	Uses Pathway's transforms to filter, join, and perform transformations on data.	**Filtering & Arithmetic:** Filtering and expression-based column computations. **Grouping & Aggregation:** Group by with reducers for live metrics. **Joins:** Snapshot-consistent merges across tables.
Temporal Operations	Uses Pathway's temporal functions to perform time-aware correlation, ordering, and windowed analytics.	**Windowing:** Tumbling, sliding, and session windows via `windowby`. **Time-Based Joins:** `asof_join` and `interval_join` for aligning late/out-of-order events.
Input & Output	Uses Pathway's connectors to ingest high-velocity observability data and export processed insights to external systems.	**Ingestion:** Kafka, filesystem, and other streaming sources. **Egress:** Kafka, PostgreSQL, Elasticsearch, cloud storage, and more.
Streaming ML Nodes	Our ML Models are also available as nodes to facilitate easy integration in the pipeline.	Online training, low-latency inference, and integration with Pathway tables.
AI & Intelligent Nodes	Combines Pathway's streaming data engine together with LLM-based reasoning to enable automated insight generation and retrieval-augmented workflows.	**Intelligent Alerting & RCA:** Triggers RCA and sends alerts. **RAG Node:** Stores documents in a Pathway Document Store exposed over MCP for agent queries.

Table 4: Core Node Categories and Their Functional Capabilities in Laminar

12.4 Agents & Streaming ML

12.4.1 Contract Parser Agent

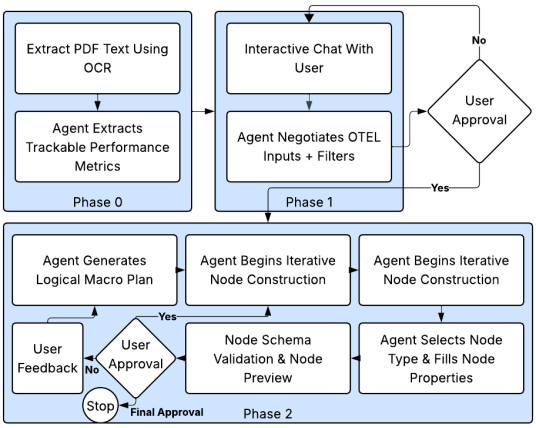


Figure 11: Contract Parser

- When the user uploads a PDF, the backend extracts the text, parses it and identifies trackable performance metrics.
- The agent launches an interactive chat session with the user, and determines which OpenTelemetry spans and filters are relevant for each metric.
- The agent generates a macro plan defining the logic for the pipeline using LLM prompts and internal planning logic.
- It iteratively constructs calculation nodes for each metric, selecting node types and filling properties.
- Each node is validated against its Pydantic schema, and the user is shown a preview before finalizing it.
- The final validated node graph is serialized as a standards-compliant flowchart.json.
- Used Claude Sonnet 4.5 for its strong overall performance and benchmarks.

Streaming & Machine Learning Applications Laminar provides predictive breach prevention through three online learning models optimized for different operational priorities. Traditional reactive monitoring alerts after thresholds breach, but Laminar's streaming ML continuously forecasts metric trajectories in advance, enabling proactive remediation before customer impact. Users select models as pipeline nodes based on their requirements:

- **TiDE - Time Series Data Encoder**: This model prioritizes **speed and resource efficiency**, since it uses a compact architecture with parallel dense layers, avoiding the sequential delays of recurrent models. With sub-millisecond inference latency and minimal memory footprint, it enables inline predictions with no observable overhead. A single server can host thousands of TiDE instances easily, making it ideal for comprehensive microservices monitoring.
- **ARF - Adaptive Random Forest**: ARF uses ensemble learning, with a forest of decision trees that vote on predictions. These trees are continuously pruned and adapted to learn recent patterns, and adapt to concept drift. ARF has **minimal memory footprint, and fast convergence** to data, which is great for cases with stricter resource constraints.
- **Mamba - State Space Model**: Mamba is a model that is **multivariate in nature and has the highest accuracy**, with a high computational need as a trade-off and uses space models to capture cross-metric dependencies. Unlike uni-variate models, it detects correlational effects better, that might be invisible otherwise (eg: when slightly elevated database query times plus reduced cache hits compound into severe latency spikes despite no single metric breaching individually). It models these correlations, with linear computational scaling. This is critical for high-stakes scenarios, where false predictions can incur severe costs. (eg: SLAs with huge financial penalties, fraud detection systems, healthcare system monitoring, etc.) This model is ideal when prediction **accuracy** outweighs resource constraints.

12.4.2 Root Cause Analysis (RCA) System

The Root Cause Analysis system helps identify *why* the platform experiences slowdowns, errors, or outages. It collects system traces, correlates events, detects patterns, and produces a structured diagnostic report. Used ChatGPT o1 in the agents for its superior structured outputs.

Working

1. **Evidence Collection**: Gathers all relevant logs, traces, and timing data around the incident window.
2. **Correlation**: Follows the request pathway to pinpoint where the issue originated.
3. **Pattern Detection**; Identifies recurring errors or failing services and checks for matching runbook solutions.
4. **Explanation**: Produces a clear, well-structured report summarising evidence and findings.

The Three Analyzers

- **Error Agent**: Analyzes error-grouped logs, detects recurring faults, traces error propagation, and identifies failing services.

- **Slowness Agent**: Locates slowest requests, identifies high-latency components, and builds hypotheses for performance degradation.
- **Downtime Analyst**: Investigates outages by inspecting pre- and post-failure logs for fatal errors or crashes.

Smart Features

- **Runbook Memory**: Reuses solutions from past incidents when similar patterns reappear.

- **12.4.3 Financial Impact Agent**

  - This agent correlates transaction logs, revenue reference data, SLA penalty clauses, etc. to understand the lost revenue from failed or mismanaged transactions or potential SLA breaches based on their duration. This calculation updates in real-time as incidents and outages progress, enabling live financial tracking on the dashboard and in the reports.

Report Generation Agent

- Triggered after RCA completion, this agent uses a LangGraph workflow based on multiple sub-agents, to analyse the incident context and the RCA agent's output, to create a structured report tailored to the target audience. Highlights of the report include summaries of the incidents, the timeline, some important triggers and error traces, and detailed charts representing affected nodes in the pipeline topology. It also contains the reccomendations based on confidence level from the rulebook consultations. These reports are stored locally using Pathway's DocumentStore, and can be indexed via vector search. The weekly report generation queries all the incident reports stored at a point of time and generates a summary document, which can be used to identify indicent patters and look at financial implications on a larger scale. This uses Gemini 2.5 for all these operations, as logs and traces need large context window that Gemini models support best.

Planner Agent

- Planner agent is the agentic system we've deployed. Any actions that require Agents, are passed to planner agents. These agents have subagents which specialize in certain tasks. The planner agent decides what needs the query have and what subagents it needs to be completed. It then queries the subagents which have tables as tool, and RAG. These subagents write SQL to query tables' postgres database, or document store. All data is then processed, and subsequent results are passed to aggregrator agent, which summarizes outputs of all subagents and answers the original query with great accuracy.

12.5 System Architecture

The system is built on a streaming, event-driven architecture that enables continuous monitoring, automated reasoning, and rapid remediation of SLA breaches. At its core, Pathway orchestrates real-time data ingestion, transformation, and model inference, ensuring that every new signal is processed with minimal latency. The architecture is modular, allowing intelligence, automation, and integrations to evolve independently.

12.5.1 Data Streaming and Ingestion Layer

Operational metrics, logs, alerts, and business events are continuously consumed from production systems and external services. Pathway normalizes and enriches this data, performing real-time feature computation such as rolling statistics, anomaly flags, and correlation indicators. This layer ensures that downstream components always operate on the freshest state of the environment.

12.5.2 Streaming ML and Detection System

Machine learning models are continuously evaluated on the incoming data to detect SLA breaches, performance degradations, or anomaly patterns. Unlike batch pipelines, the models are updated on-stream, enabling sub-second recognition of any critical shift in service behavior. This enables proactive root cause identification rather than post-incident diagnostics.

12.5.3 Agentic Decision Layer

An autonomous agent interprets detected incidents, reasons through potential hypotheses, and determines the optimal response strategy. It performs multi-step planning, selects relevant tools or APIs, and triggers targeted actions such as diagnostic workflows, report generation, or automated mitigations. The agent preserves context across steps to maximize resolution accuracy and minimize false triggers.

12.5.4 Action and Communication Interfaces

Once a resolution path is chosen, the system executes the required operations—ranging from orchestration API calls to ticket creation or remediation scripts. Insights, RCA summaries, and SLA compliance reports are delivered to teams through their operational channels such as Slack or dashboards. This ensures that human stakeholders are informed only when necessary, while routine response remains automatic.

12.6 Guardrails

To ensure the integrity of the Agentic System, all the incoming requests are passed through the **MCPSecurityGateway**. This gateway functions act as a centralized firewall, enforcing a Defense-in-Depth strategy. A request must successfully pass through the four-stage validation pipeline before it is processed.

Layer 1 & 2: Access Control and Network Integrity

The first line of defence focuses on *who* is making the request and *where* it is going. This layer filters out unauthorized access and structural attacks before any processing occurs.

- **Availability Protection (Rate Limiting):** The system checks the number of incoming requests. Users who exceed the defined quota are temporarily blocked to prevent Denial-of-Service attacks and ensure stable system performance.
- **Network Boundary Protection (SSRF Check):** To ensure that the system cannot be misused as a proxy to target internal networks, the destination URL is validated and Only legitimate, external, HTTPS endpoints are allowed, while the attempts to reach private IPs or internal services are blocked.
- **Authorization & Identity (Role & Ownership Checks):**
  - *Role Verification:* The system enforces the Principle of Least Privilege by verifying whether the user's role actually allows the action before letting them perform it, so they only get the access they truly need.
  - *Ownership Verification:* To prevent unauthorized access, the system confirms that the user owns the resource they are attempting to access.

Layer 3: Content Analysis & Intelligent Scanning

Once the request's legitimacy is approved, the payload is provided to inspection using natural language processing and pattern-based analysis. Currently, all the layers have been implemented, but layer 3 is actively used now; it is used independently on inputs(such as logs, traces and errors) before sending to agents (e.g. RCA agent).

- **Prompt Injection:** We used a Deep learning model(protectai/deberta-v3-base-prompt-injection-v2) to analyze the text input to detect jailbreak attempts designed to bypass system safeguards.
- **Data Loss Prevention (DLP):**
  - *PII Scanning:* We used Microsoft Presidio Spacy that automatically scans for personally identifiable information to ensure privacy and compliance.
  - *Secrets Detection:* We used Heuristic Regex patterns to detect secrets such as API keys , tokens etc(AWS keys, GitHub tokens, Slack tokens).
- **Obfuscation Detection:** The system checks for invisible or non-standard Unicode characters often used in obfuscation attacks to hide malicious instructions.
- **Human-in-the-Loop (HITL):** When the scanners detect a risky but ambiguous pattern, the request is paused and routed to a human reviewer for manual approval.

Layer 4: Execution & Process Integrity

The final layer governs the safe execution of the request and apply strict post processing.

- **Data Integrity Verification:** When receiving data from external tools, the system validates digital signatures (HMAC) to ensure that the information has not been tampered with during transit.
- **Output Sanitization:** Before returning results to the user, the system removes potentially malicious elements such as embedded scripts or unsafe HTML to prevent Cross-Site Scripting (XSS).
- **Session Hygiene:** To reduce session hijacking risks, the system applies a clean-slate policy, wiping session context and rotating security tokens immediately after the workflow completes.

12.7 User Interface

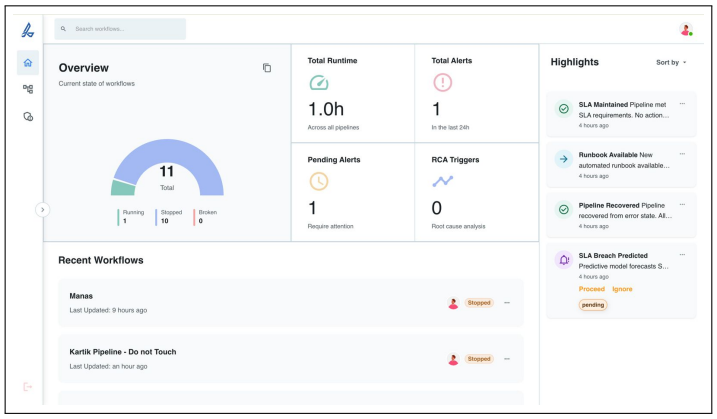


Figure 12: Overview Page

The left section of the overview page provides a general glimpse of the data in the form of a semi-circle gauge chart, along with KPIs of total runtime, total alerts, pending alerts and RCA triggers, followed by the recent workflows section where the user can access the recently updated pipelines. On the left side of the page, the highlights section displays notifications and alerts to ensure that the human in the loop is notified whenever required. The user can also filter them on various parameters like type, information, error, warning etc.

On the admin page, the user with the admin role, can select any of the four metrics on the top left to visualise the data through a graph. Below this, to the left is a list of all workflows, which enables the admin to view and edit all workflows. On the right is Members section, where the admin can see the details of all members registered on the platform.

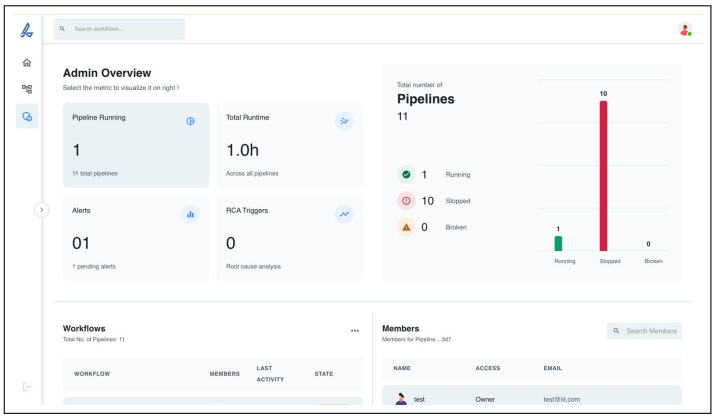


Figure 13: Admin Page

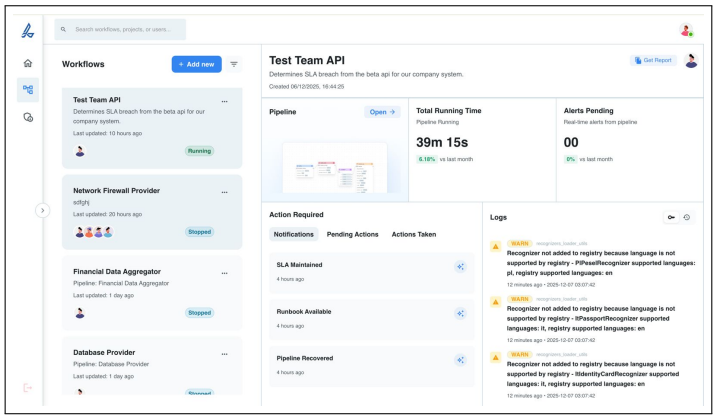


Figure 14: Workflow Page

The Workflows Page gives users a focused view of their individual workflows. A list of all workflows appears on the left, and selecting one reveals its detailed information on the right. Users can generate a workflow report, view metrics such as total running time and pending alerts, and compare current statistics with those from the previous month. The Actions Required section further helps users stay up to date by organizing notifications, pending actions, and completed actions into separate tabs. Below this, the Logs and Version History section allows users to track workflow logs, monitor previous versions, and identify who made changes and when.

When creating a new workflow, users can click on Add New Workflow, which opens a popup that leads to an interactive canvas. On this canvas, users can add nodes from various node collections, configure them, and also make use of the integrated AI chatbot for assistance. They can activate or deactivate the workflow, run or stop it, and access the Runbook, which includes predefined actions and alerts. This workflow creation interface allows users to build and manage automation pipelines with clarity and flexibility.

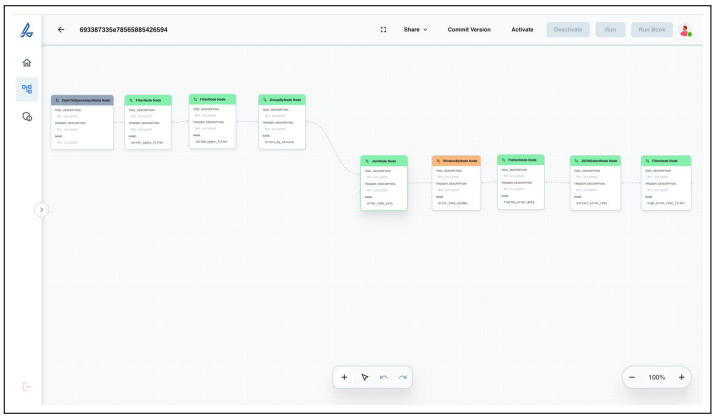


Figure 15: Canvas