ISTANBUL
SOFTWARE
TESTING
CONFERENCE

Surviving the Avalanche

Hayrullah Öztürk
QA Manager @Logiwa

#ISTC2026

Zero Defects at 8x Load with AI-Driven Testing

About Me

- QA Manager @Logiwa WMS: Leading End-to-End Test Automation & Quality Strategies.
- 10+ Years in Industry & 5+ Years Engineering Leadership: Specializing in scaling QA infrastructures.
- The Hyper-Growth DNA: Built testing frameworks from the ground up at Digiturk, Getir and Logiwa .
- Tech Instructor @LabaTurkey: Bridging the gap between traditional Java automation and AI-driven testing.

#ISTC2026


AI
YEAR 1
YEAR 2
YEAR 3
COMPONENT-LEVEL STRESS
• Breaking individual endpoints.
• Found infrastructure limits, missed user complexity.
REALISTIC WORKLOAD MODELING
• Simulating complex user journeys.
• Stable E2E tests, but too static for unpredictable peaks.
AI-DRIVEN TRAFFIC ANALYSIS
• Dynamic test intensity based on AI-analyzed production logs.
• Predictive scaling.

Static Scripts vs. Real Chaos

- The Illusion of the "Happy Path"
- Chaos Like The "Black Friday"
- Multi-Tenant Blind Spots
- Manual Bottleneck

PIVOT

Stop Guessing. Let the Data Write the Tests.

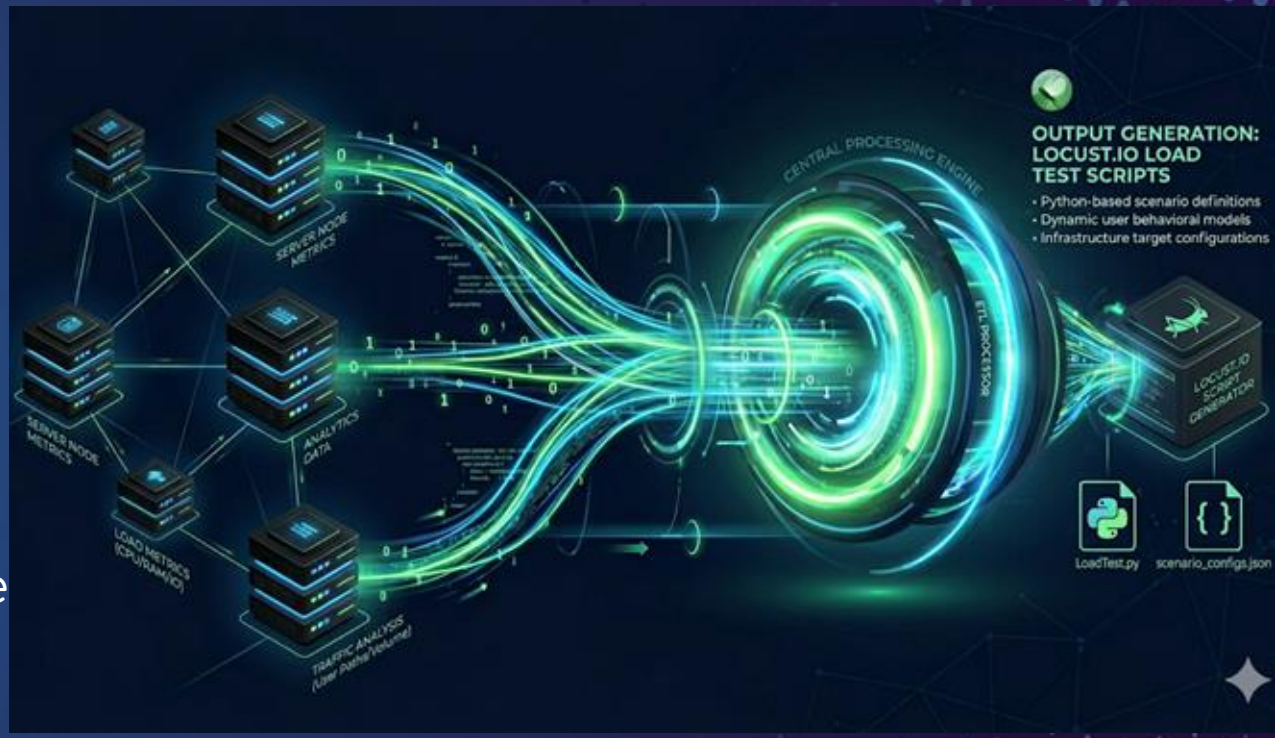
Mining the Chaos:
Let Data Write the Tests

- From Guesswork to Data-Driven:
  Stop guessing start analyzing
- Mining Production Logs:
  Analyze Data with ML models
- Finding the Hidden Patterns:
  No manual scripts
- Dynamic Test Generation:
  Real-world usage patterns into test scenarios.

#ISTC2026

Building the Engine: From Logs to Locust

- Data Ingestion:
  New Relic, Kibana, and DB monitors.
- The Translation Layer:
  AI models generating weighted user journeys
- The Execution Engine:
  A fully custom, auto-scaling load infrastructure built with Python and Locust.io.
- The Strategy:
  Feeding AI-generated load profiles directly into our clusters.



Architecture: From Logs to Locust

Production Logs

Ingesting telemetry from New Relic & Kibana

Synthetic Setup

Privacy first: Zero-data leak account mirroring

AI Modeling

Generating weighted, multi-tenant user journeys

Locust Engine

Python-based dynamic scaling execution

Extreme Over Testing: 3x The Expected Avalanche

Multiplying the Predicted Avalanche

To ensure absolute zero defects, we took our predicted 800% peak traffic and multiplied it by 3 in our simulations.
Our custom Locust engine simulated a catastrophe far worse than any possible real-world peak.

The Results: Black Friday Peak

Metric (3-Hour Window)	Target / SLA	Actual Outcome	Status
Total Requests Processed	1 M (Forecast)	1.82 Million	Success
Median Response Time	< 100 ms (SLA)	35 ms	Outperformed
User-Facing Impact	Zero Downtime	0 Impact	Flawless
Failed Requests	< 0.1%	1 Isolated Fail	Acceptable

The Next Frontier: Real-Time Load Orchestration

- Phase 1: Pre-Configured Precision:
  Simulated massive peaks with static AI profiles.
- Phase 2: The Orchestrator:
  Moving from automated setup to real-time execution control.
- Dynamic Adaptation:
  Test intensity with live system feedback.
- The Ultimate Goal:
  Continuous, autonomous resilience testing that breathes with the system.

#ISTC2026

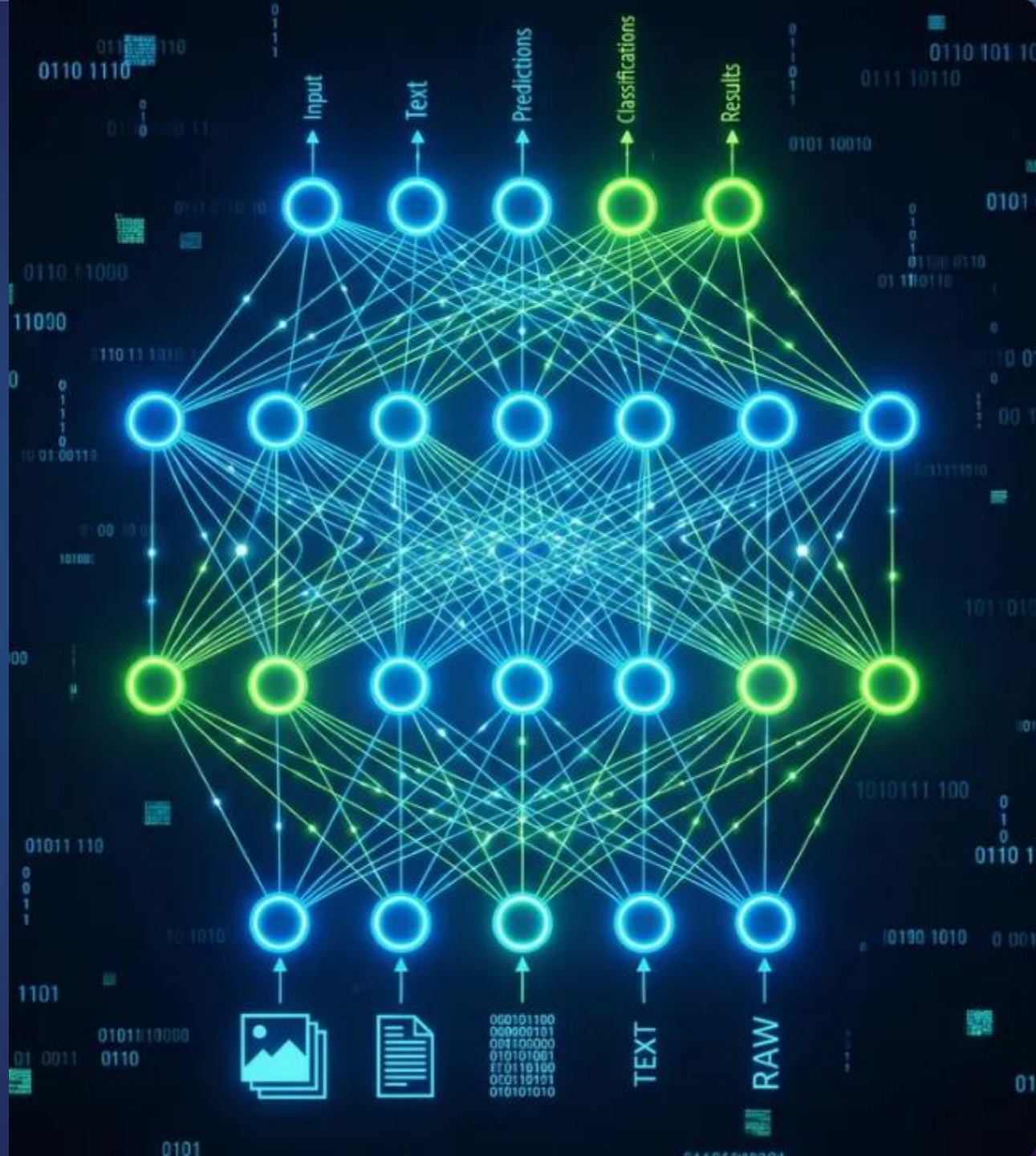
What to Apply Tomorrow?

- **Audit Production Patterns:**
  Stop guessing; start mining logs for real user journeys.

- Embrace Synthetic Mirroring:
  Don't mask sensitive data; replicate the weights in dummy setups.

- Scale Your Own Engine:
  Use a load testing library for flexible, scriptable load execution.

#ISTC2026



The Future of Quality Engineering

AI shouldn't think for you; it should be the weapon that executes the 'impossible' scenarios your imagination creates.

The Future of Quality Engineering

Don't just manage the flow. Hear its heartbeat. Be the ultimate sense of your system through AI-powered insight.

The Future of Quality Engineering

We aren't here to stop the flow,
but to empower the speed;
quality is the engine, not the
brake.

Thank You!

Surviving the Avalanche Together

Hayrullah Öztürk
Software QA Manager @Logiwa

LOGIWA

Questions?

#ISTC2026