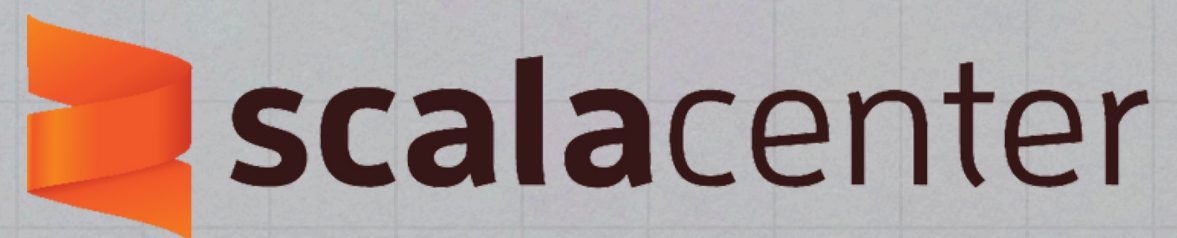


LLM4S: BUILD RELIABLE AI SYSTEMS IN SCALA

OAISYS AI PRACTITIONERS CONFERENCE 2025

Presented by Kannupriya Kalra & Shubham Vishwakarma



Why this matters

The problem:

- Most LLM frameworks force you to choose
- Fast iteration OR type safety
- Production readiness OR developer velocity

What if you didn't have to choose?

- LLM4S brings type-safe, functional AI development to the JVM. Reliability is built into the type system.

AI hype vs Reality

The data doesn't lie:

- MIT report: 95% of generative AI pilots at companies are failing
- 88% of AI pilots fail to reach production
- Steep drop: 80% investigated → 50% piloted → 5% successfully implemented

Why pilots fail:

- Hallucination and lack of grounding Scaling and latency issues
- Management gaps: observability, measurability, feedback cycles
- The shift: From "cool prototypes" → to "critical systems"

What is LLM4S?

Mission: Type safety, functional programming, and JVM reliability for AI development

A framework for building LLM applications in Scala. Works directly with your existing systems.

A reliable AI system = LLM + Context + Data + Tools + Observability + Governance +

Feedback Cycles

- Context grounds responses
- Data ensures truth
- Tools enable action
- Observability tracks behavior
- Governance builds trust
- Feedback cycles allow optimization



5 Pillars of Reliability

LLM4S design philosophy:

1. Composable modules - RAG, agents, tracing, tools
2. Type safety - Catch prompt and context errors at compile time
3. Tracing and observability - Understand, debug, and optimize
4. Multi-provider support - OpenAI, Anthropic, Gemini, Ollama, Azure
5. Scalability - Plug-and-play architectures with consistent APIs

Reliability isn't post-facto testing → it's design.

Building AI like software: structured, measurable, and predictable.

Core Capabilities

Multi-provider support

- OpenAI, Anthropic, Azure, Ollama, OpenRouter - One interface, zero lock-in

Type safety first

- Compile-time error checking - No runtime surprises

Functional error handling

- Either-based composition - No exceptions thrown

Agent framework

- Single and multi-agent workflows - Tool calling with auto-generation

Multimodal support

- Text, images, voice - RAG with built-in retrieval

Why join this Open-source project?

- First AI framework in Scala ecosystem
 - Be part of something historic
 - Shape the future direction
- Learn from the best
 - Staff engineers, architects, engineering leaders mentor you
- Weekly live coding sessions
 - Real learning, not just tutorials
- Google Summer of Code funding
 - Get paid to contribute
- Global community

The Team

Core maintainers:

Rory Graves - Lead maintainer and chief architect

- Staff engineer, Scala compiler contributor

Kannupriya Kalra - Lead maintainer and community lead

- Engineering leader, Scala Center mentor

Atul S. Khot - Core contributor and design architect

- Smart constructors, functional error handling, type class design

Vitthal Mirji - Core contributor and Low-level design

- Error handling hierarchy, Type class designs, FP patterns

Dmitry Mamonov - Core contributor and integration specialist

Every Sunday: Live Coding

When: Sunday 9am London time

Where: LLM4S Discord - llm4s-dev-hour channel

Calendar: Weekly Luma invite

What happens:

- Community votes on features
- Mob programming live
- Pair programming with staff engineers
- Design architecture collaboratively
- Ship features to production

First in the Scala Ecosystem

Career impact:

- Portfolio standout: "Contributed to the first Scala AI framework"
- Early adopter advantage: Shape the ecosystem before mainstream
- Unique expertise: Be the Scala plus AI expert
- Conference talks: Present your contributions at major events
- Hiring magnet: Companies need this skillset now

Technical impact:

- Influence design decisions: Your ideas shape the future
- Learn cutting-edge patterns: Before they're documented
- Build reputation: Known in Scala and AI communities
- OSS credibility: Recognized by Scala Center, Lightbend, Typelevel

Progress & Momentum

2025 achievements:

- Selected for Google Summer of Code 2025
- 4 GSoC contributors funded by Google
- Cross-compilation support (Scala 2.13 and 3.x)
- Multi-provider abstraction complete
- Agent framework with tool calling
- Langfuse tracing integration
- Docker-based secure execution

Conference presentations:

- Bay Area Scala, Scala India, Functional World, Dallas Scala Enthusiasts
- Scala Days, GSoC Lightning Talks, ICFP/SPLASH

v1.0 production scheduled release - December 2025

Production Ready

What production ready means:

Type safety:

Compile-time error checking, Exhaustive pattern matching, Smart constructors

Error handling:

Either-based composition, Recoverable vs non-recoverable errors, Retry strategies

Observability:

Langfuse tracing integration, Token usage tracking, Performance metrics

Scalability:

Async operations with Futures, Context management, Connection pooling

Security:

Docker Container Isolation, API key Management, Secure Workspace Execution

Global Community

Contributors across the globe: USA, United Kingdom, Poland, India, Turkey, Germany, Switzerland, Singapore

GSoC 2025 contributors:

Elvan Konukseven (Switzerland) - Agentic toolkit

Gopi Trinadh Maddikunta (India) - RAG in a box

Anshuman Awasthi (India) - Multimodal support

Shubham Vishwakarma (India) - Tracing system

Join the community:

Discord: <https://discord.gg/4uvTPn6qww>

GitHub: <https://github.com/llm4s/llm4s>



THANK YOU & GET STARTED!

Join us today:

Discord: <https://discord.gg/4uvTPn6qww>

GitHub: <https://github.com/llm4s/llm4s>

Weekly dev hour links:

[https://luma.com/calendar/cal-](https://luma.com/calendar/cal-Zd9BLb5jbZewxLA)

[Zd9BLb5jbZewxLA](https://luma.com/calendar/cal-Zd9BLb5jbZewxLA)

See you in dev hours every Sunday 2 30pm IST.

Let's build the future of AI together.



Connect with us :)