

Guardrails as a Service

CSL7090 - Project , IIT Jodhpur

Group No: 8

IIT Jodhpur

November 16, 2025



Pradeep Annepu (M25AI1109) Rajesh Akula (M25AI1048)
Amarendra Chakravarthi (M25AI1082) Anirudh Reddy (M25AI1131)

Outline

- 1 Introduction
- 2 Architecture
- 3 Domain Model
- 4 Technology Stack
- 5 Testing
- 6 Quality attributes



Guardrails as a Service

- AI safety and policy enforcement platform
- Microservices architecture for scalability
- Real-time evaluation of AI prompts and responses
- Policy-based governance for AI applications
- Comprehensive audit logging and monitoring



Core Capabilities:

- Policy management
- Real-time evaluation
- Lora based inference Model
- Audit logging
- Rate limiting

Technical Features:

- RESTful API
- Event-driven architecture
- Caching with Redis
- PostgreSQL persistence
- Kafka message bus



Five Microservices

① API Gateway (Node.js) - Port 3000

- Authentication & Authorization
- Request routing
- Rate limiting

② Evaluation Service (Node.js) - Port 3001

③ Policy Service (Node.js) - Port 3002

④ Audit Service (Node.js) - Port 3003

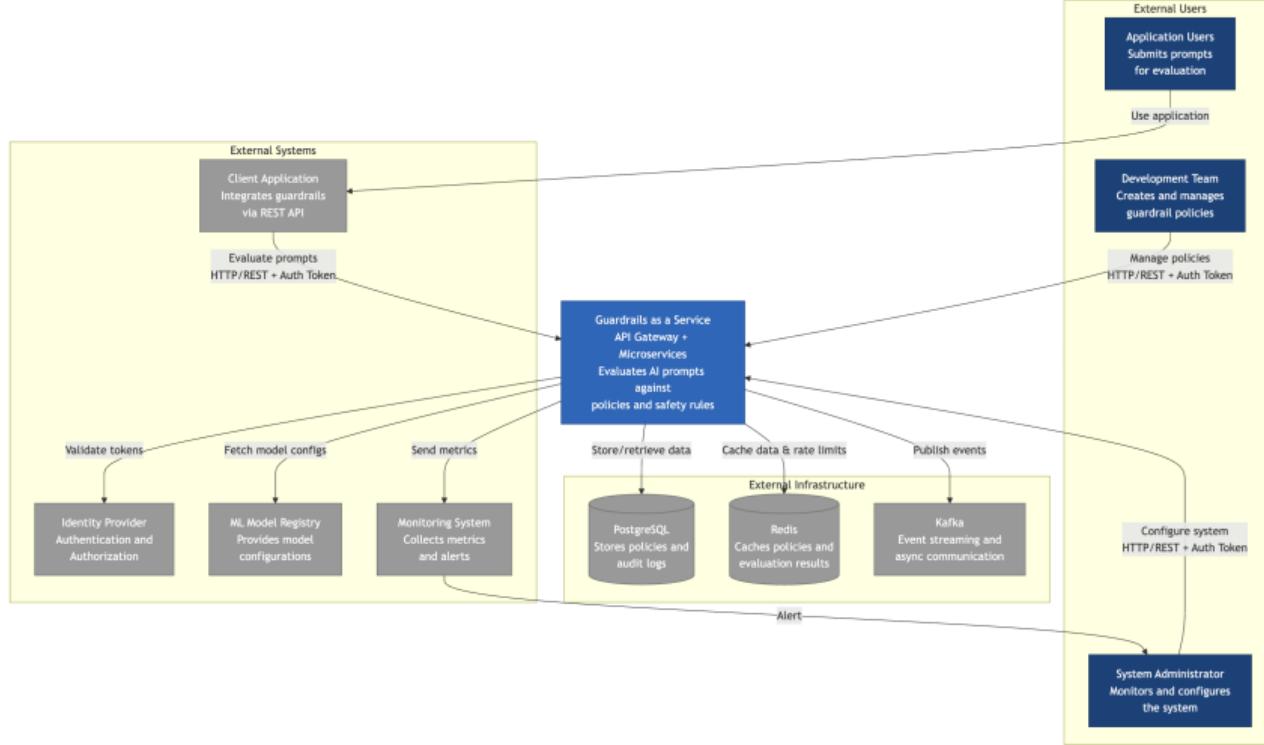
⑤ Model Inference Service (Python/FastAPI) - Port 8000



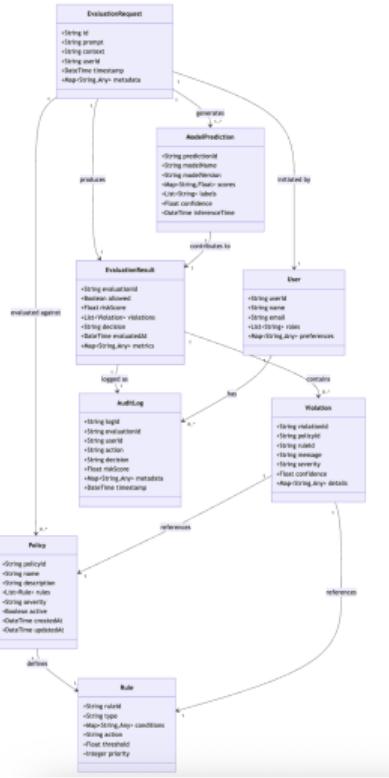
High-Level Architecture



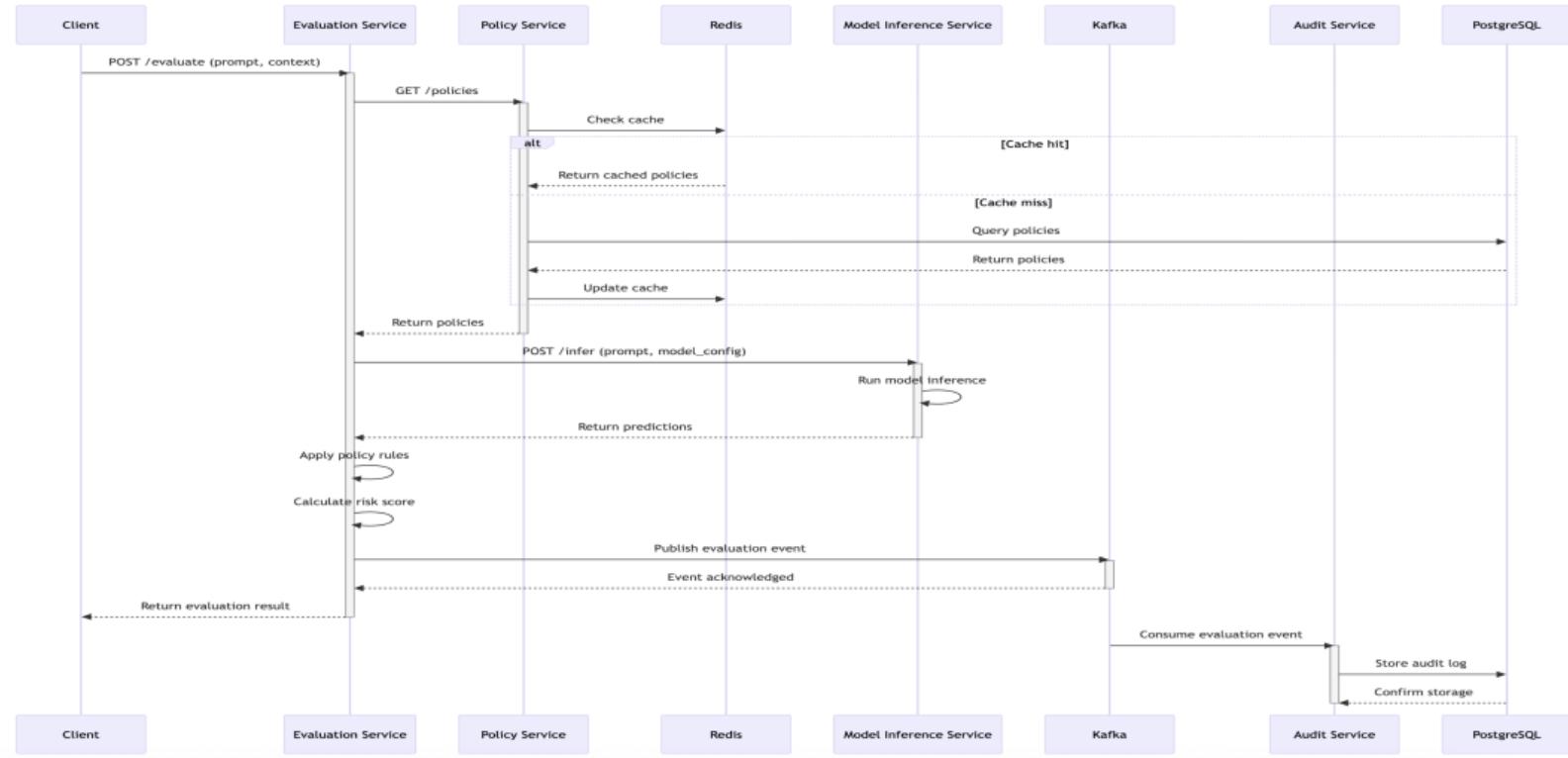
C4 System Context



Core Domain Entities



Sequence Diagram: Evaluation Flow



Technology Stack

Backend Services:

- Node.js ($\geq 18.0.0$)
- TypeScript
- Express.js
- Python (≥ 3.8)
- FastAPI

Infrastructure:

- PostgreSQL (persistence)
- Redis (caching)
- Apache Kafka (messaging)

Development Tools:

- npm ($\geq 8.0.0$)
- pip
- ESLint
- Prettier

Testing & Monitoring:

- Jest (unit testing)
- k6 (load testing)
- Monitoring integration



Load Testing with k6

Prerequisites:

```
1 # macOS
2 brew install k6
3
4 # Windows
5 choco install k6
```

Running Load Tests:

```
1 cd load_test
2 k6 run load_test.js
3
4 # With web dashboard
5 k6 run --out web-dashboard load_test.js
```



Load Test Configuration

Default Load Test Pattern

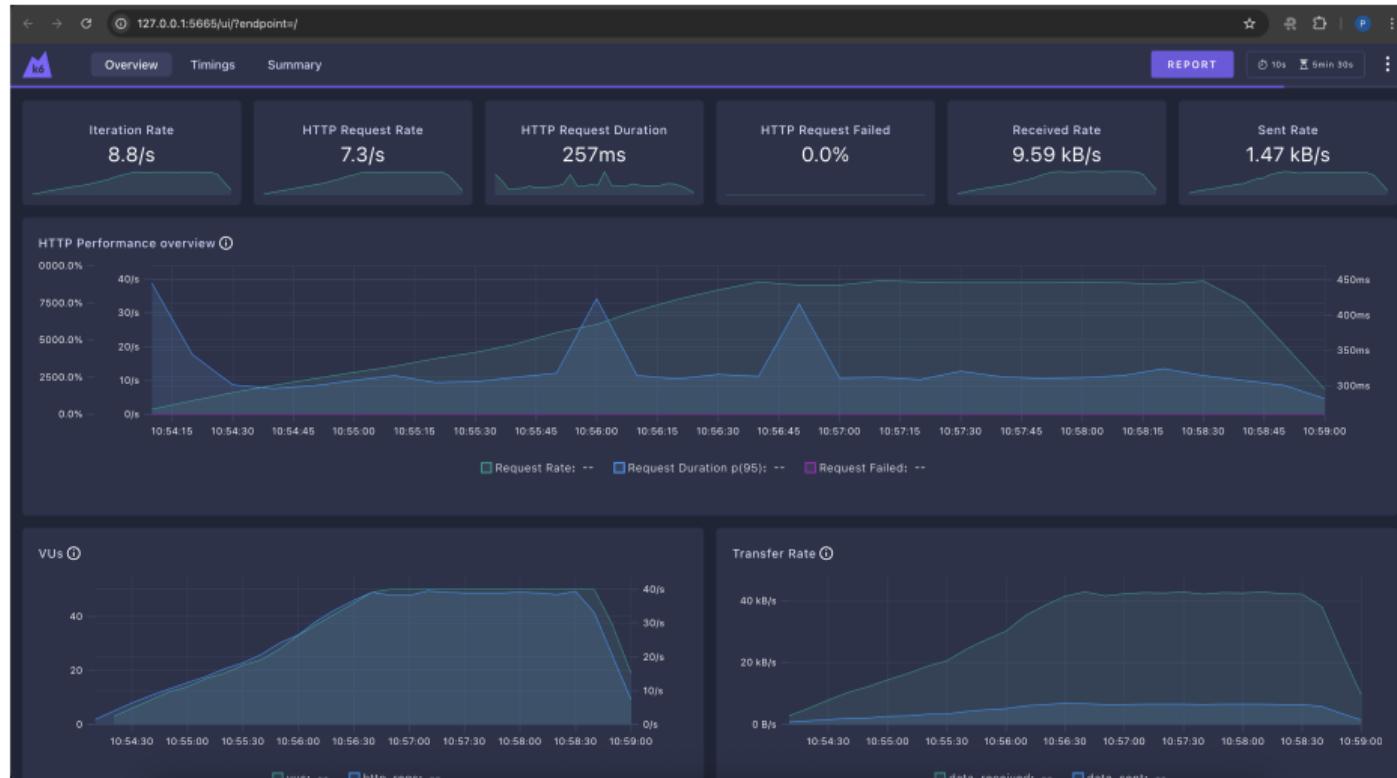
- 30s: Ramp up to 10 users
- 1m: Ramp up to 25 users
- 1m: Ramp up to 50 users
- 2m: Maintain 50 users (peak load)
- 30s: Ramp down to 0 users

Configuration Notes

- Update host link to API Gateway URL
- Include authentication tokens
- Ensure all services are running before testing



Load Test Results



Quality attributes

① Security

- Centralized authentication and authorization via API Gateway
- Rate limiting to prevent abuse

② Scalability

- Microservices architecture allows independent scaling
- Redis caching to reduce latency
- Kafka for asynchronous communication

③ Flexibility

- Easy to add new policies and rules
- Modular services for future enhancements



Thank You!

GitHub: github.com/pradeepannepu/guardrails-as-a-service

