FINAL TECH STACK — ENTERPRISE AI BACKEND PLATFORM

---

1. CORE PROGRAMMING LANGUAGES & RUNTIMES

Python 3.11+

Role: Primary AI and data execution language
Why it exists in the stack:

- Model training, inference logic, feature engineering
- Data pipelines and ML workflows
- Industry-standard for ML/AI ecosystems

Key characteristics:

- Massive ML ecosystem
- Fast iteration
- Strong community support
- Production-safe when isolated behind services

Java 17

Role: Enterprise-grade backend and workflow execution
Why it exists:

- Deterministic execution for business logic
- Long-running services

- Strong typing and JVM stability

Used for:

- Decision engines
- Policy evaluation
- Workflow coordination
- High-reliability backend services

Go

Role: Infrastructure and high-concurrency services
Why it exists:

- Lightweight binaries
- Excellent concurrency model
- Ideal for control-plane services

Used for:

- Streaming consumers
- Infra utilities
- Performance-critical microservices

---

2. EVENT INGESTION & STREAMING

Apache Kafka

Role: Central nervous system of the platform

Purpose:

- Ingest high-volume real-time data
- Decouple producers and consumers
- Enable replay, backfilling, and parallel processing

Key capabilities:

- Partitioned topics
- Exactly-once semantics
- Schema enforcement (Avro / Protobuf)

Kafka Connect

Role: Managed data movement
Purpose:

- CDC from databases
- Sink data into object storage
- Eliminate custom ingestion code

Redis Streams

Role: Lightweight asynchronous task queue
Purpose:

- Retry mechanisms
- Short-lived buffering
- Low-latency async workflows

---

3. DATA STORAGE (POLYGLOT STRATEGY)

Amazon S3 (or equivalent)

Role: Immutable system of record

Purpose:

- Store raw, cleaned, and aggregated datasets
- Persist training data and model artifacts
- Enable reproducibility and auditability

Why object storage:

- Cheap
- Durable
- Infinitely scalable
- Replay-friendly

PostgreSQL

Role: Transactional and metadata database

Stores:

- Workflow state
- Configurations
- Model metadata
- Feature definitions

Why PostgreSQL:

- Strong consistency
- Mature ecosystem
- Operational reliability

ClickHouse

Role: High-performance analytical database

Used for:

- Feature aggregations
- Event analytics
- Real-time dashboards
- Sub-second queries on large datasets

TimescaleDB

Role: Time-series data management

Used for:

- Metrics
- Temporal signals
- Trend and seasonality analysis

---

4. FEATURE ENGINEERING & DATA INTELLIGENCE

Feast (Feature Store)

Role: Feature consistency enforcement

Purpose:

- Ensure training and inference use identical features
- Maintain feature lineage
- Prevent training-serving skew

Components:

- Offline store (training)
- Online store (inference)

---

5. AI / ML DEVELOPMENT STACK

PyTorch

Role: Core deep learning framework

Used for:

- Neural networks
- LLM fine-tuning
- Custom model architectures

Hugging Face Transformers

Role: LLM and NLP ecosystem

Used for:

- Pre-trained language models
- Fine-tuning
- Retrieval-Augmented Generation (RAG)

XGBoost / LightGBM

Role: Classical ML models

Used for:

- Scoring systems
- Risk models
- Structured-data prediction

Why included:
Most production ML problems do not require deep learning.

---

6. DISTRIBUTED TRAINING & EXPERIMENTATION

Kubernetes

Role: Compute orchestration backbone

Purpose:

- Schedule training and inference jobs
- Isolate workloads
- Auto-scale resources

Kubeflow

Role: ML workflow orchestration

Used for:

- Training pipelines
- Experiment tracking
- Model registry integration

Ray

Role: Distributed compute framework

Used for:

- Parallel training
- Hyperparameter tuning
- Scalable experimentation

---

7. MODEL REGISTRY & GOVERNANCE

Model Registry (Kubeflow / MLflow-style)

Role: Model lifecycle control

Stores:

- Model versions
- Training metadata
- Feature lineage
- Approval status

Why mandatory:
Without this, rollback, audits, and trust are impossible.

---

8. MODEL SERVING & INFERENCE

KServe

Role: Kubernetes-native model serving

Capabilities:

- REST/gRPC inference
- Auto-scaling
- Canary deployments

NVIDIA Triton Inference Server

Role: High-performance inference engine

Used for:

- GPU-accelerated inference
- Multi-model serving
- Low-latency predictions

---

9. DECISION INTELLIGENCE & EXPLAINABILITY

Rule Engine (Drools or Custom DSL)

Role: Deterministic decision logic

Used for:

- Policy enforcement
- Thresholding
- Guardrails around ML outputs

OPA (Open Policy Agent)

Role: Policy-as-code

Purpose:

- Access control
- Decision constraints
- Governance enforcement

SHAP / LIME

Role: Explainability layer

Used for:

- Feature attribution
- Decision transparency
- Regulatory justification

---

10. WORKFLOW ORCHESTRATION

Apache Airflow

Role: Batch pipeline orchestration

Used for:

- Feature pipelines
- Training jobs
- Scheduled tasks

Temporal

Role: Stateful workflow engine

Used for:

- Long-running AI workflows
- Exactly-once execution
- Failure recovery

---

11. BACKEND SERVICES & APIs

FastAPI

Role: AI-facing API layer

Used for:

- Inference APIs
- Internal AI services
- High-performance async endpoints

Spring Boot

Role: Enterprise backend services

Used for:

- Core business logic
- Workflow coordination
- Policy execution

---

gRPC

Role: Internal service communication

Purpose:

- Low latency
- Strong contracts
- Efficient serialization

---

12. SECURITY, IDENTITY & SECRETS

Keycloak

Role: Identity and access management

Provides:

- OAuth2 / OIDC
- RBAC
- Service identity

HashiCorp Vault

Role: Secret and key management

Used for:

- API keys
- Certificates
- Database credentials

---

13. OBSERVABILITY & AI RELIABILITY

Prometheus

Role: Metrics collection

Grafana

Role: Visualization and dashboards

OpenTelemetry

Role: Unified observability instrumentation

ELK / Loki

Role: Centralized logging

AI Monitoring

Capabilities:

- Data drift detection
- Concept drift detection
- Performance decay alerts
- Bias monitoring

---

14. CI/CD & INFRASTRUCTURE

Docker

Role: Containerization

Helm

Role: Kubernetes deployment management

GitHub Actions

Role: CI/CD pipelines

Used for:

- Testing
- Model promotion
- Automated deployment