📘 Technical Architecture Document

Product: GravitasETRM
Version: v1.0
Prepared by: Engineering Team
Date: July 2025

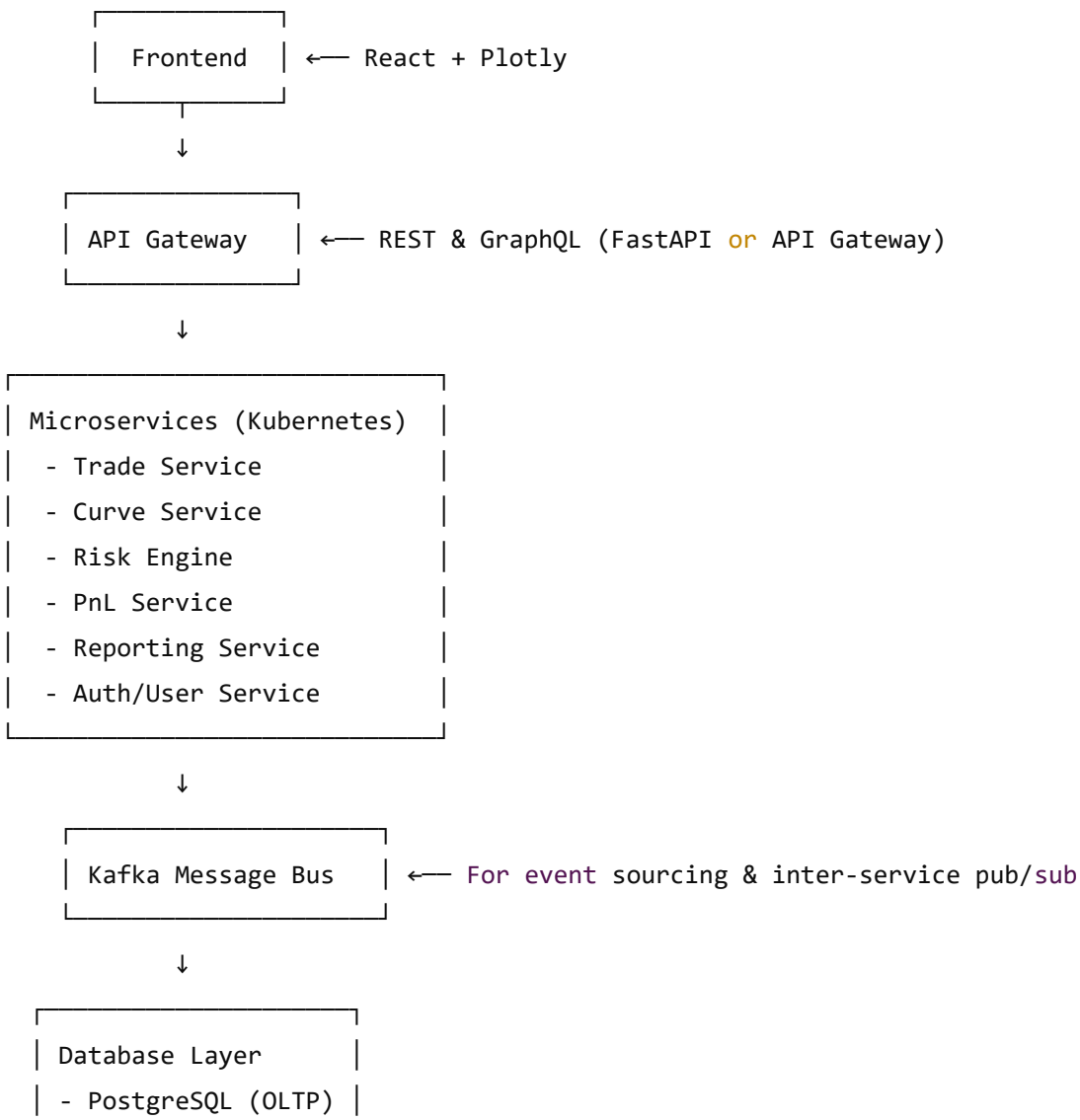
---

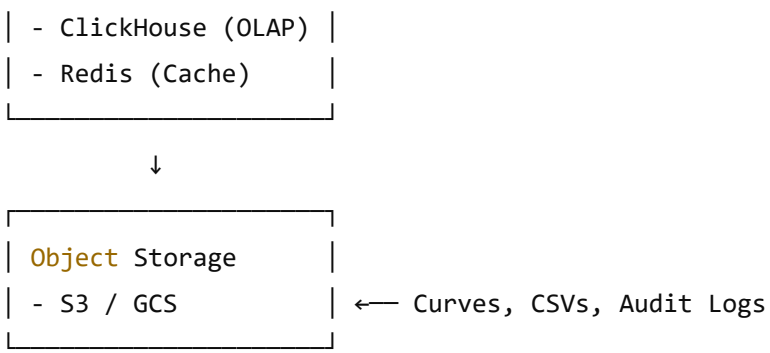
1. 🧭 Objective

To define the **technical architecture** and system components of **GravitasETRM**, a cloud-native, modular Energy Trading and Risk Management system that supports real-time deal capture, market risk, curve management, and settlements for power, gas, oil, and emissions trading.

---

2. 🧱 High-Level Architecture





3. ⚙️ Technology Stack

Layer	Technology
Frontend	React.js, Plotly, Tailwind CSS
Backend APIs	FastAPI (Python), gRPC (Go)
Event Streaming	Kafka, Debezium (CDC), Redis
Compute & Jobs	Kubernetes (K8s), Argo Workflows
Databases	PostgreSQL, ClickHouse, Redis
Storage	S3 (AWS) / GCS (Google)
Risk Engine	Python (NumPy, Pandas, PyPortfolio), Rust (optional for performance)
Auth	Keycloak / Auth0 (OAuth 2.0 + RBAC)
Monitoring	Prometheus, Grafana, Loki, OpenTelemetry
CI/CD	GitHub Actions, Helm, ArgoCD

4. 📂 Key Components

4.1. Trade Service

- Handles CRUD operations for trades
- Supports vanilla and structured deals

- Stores metadata, curve references, counterparty
- Emits events to Kafka on trade insert/update

4.2. Curve Management

- Manages forward/volatility curves
- Stores versioned curve snapshots (Parquet + S3)
- Performs interpolation, smoothing, validation
- Integrates with external feeds (ICE, CME, Refinitiv)

4.3. Risk Engine

- Calculates VaR (Historical, Parametric)
- Computes PnL, Greeks (Delta, Gamma, Vega)
- Scenario stress testing (user-defined shocks)
- Exposes endpoints for intraday reports

4.4. PnL Service

- Recomputes PnL whenever trade/curve changes
- Supports snapshotting at end-of-day
- Provides intraday PnL breakdowns by desk, book, trader

4.5. Reporting Service

- Connects to ClickHouse for dashboard queries
- Supports ad-hoc SQL access and data exports
- Integrates with Metabase/Tableau/Power BI

4.6. Auth & User Management

- Uses OAuth2.0/OIDC for secure login
- Role-based access control (RBAC)
- Logs all user activity and API usage

---

5. 🗄️ Data Architecture

5.1 PostgreSQL (OLTP)

Used for transactional systems:

- Trades
- Curve metadata

- User accounts and roles
- Audit logs

5.2 ClickHouse (OLAP)

Used for real-time reporting:

- Time-series PnL data
- Risk results (VaR, sensitivities)
- Curve history and snapshots

5.3 Object Store (S3)

- Curve files
- Backtest results
- CSV exports
- Audit reports

---

6. 🔄 Data Flow

New Trade Flow:

1. User submits trade via UI → API Gateway → Trade Service
2. Trade Service validates and persists to PostgreSQL
3. Trade Service emits event to Kafka (`trade.created`)
4. Risk Engine & PnL Service listen and trigger calculations
5. Results are stored in ClickHouse
6. UI dashboards update in real-time

---

7. 🔐 Security & Compliance

- OAuth2 + RBAC per module
- Field-level encryption for PII
- Audit logs for every action (immutable S3)
- GDPR and EMIR/REMIT export API
- SOC2 and ISO 27001 ready deployment pipeline

---

8. 🧪 Testing & Quality

Type	Tools
Unit Testing	PyTest, GoTest
Integration	Postman, Docker Compose
Load Testing	Locust, K6
E2E Testing	Playwright
Monitoring	Prometheus + Grafana
Tracing	OpenTelemetry

---

9. 📈 Scalability Strategy

- All services are containerized and deployed on Kubernetes
- Kafka ensures asynchronous processing
- ClickHouse is horizontally scalable for fast reporting
- Redis cache for recent trades and PnL
- S3 used for infinite historical storage

---

10. 🔄 DevOps & Deployment

- **CI**: GitHub Actions for lint, test, build
- **CD**: ArgoCD to deploy Helm charts
- **Infra**: Terraform scripts to provision cloud infra
- **Envs**: Dev, QA, UAT, Prod (with config maps)

---

11. 🔄 External Integrations

System	Integration Type
Market Data (ICE, Refinitiv)	REST, SFTP
ERP (SAP, Oracle)	JSON/CSV export

System	Integration Type
Scheduling Systems (TSO)	REST APIs
Risk Models (3rd Party)	Plug-in based