Below is a **clear, enterprise-grade technology stack definition** for **governance, data, application, infrastructure, security, and program execution** for a **large healthcare multi-product platform**.

This stack is **audit-ready, scalable, cloud-agnostic**, and aligned with everything you've built so far (EA, SA, App, Data, Security, OKRs, guardrails).

Enterprise Technology Stack (Reference)

Scope: Healthcare multi-product platform (clinical, member, provider, analytics, AI)
Design goal: Standardized, self-service, secure, observable, governable

---

1️⃣ Governance & Architecture Stack

Architecture & Design

- **Confluence** – Architecture docs, ADRs, standards
- **Draw.io / Lucid / Archi** – Diagrams (C4, ArchiMate)
- **LeanIX / Ardoq** – EA repository, capability maps
- **ServiceNow APM** – Application portfolio management

Governance Automation

- **OPA (Open Policy Agent)** – Policy-as-code
- **HashiCorp Sentinel** – Terraform policy
- **Backstage** – Developer portal + standards
- **Jira** – Architecture reviews, exceptions
- **GitHub** – ADR versioning

2️⃣ Data & Analytics Stack

Ingestion

- **Kafka / Confluent / MSK**
- **Kinesis / Event Hubs**

- **Fivetran / DMS / Debezium**
- **API Gateway ingestion**

Storage

- **Lakehouse (Delta / Iceberg / Hudi)**
- **S3 / ADLS / GCS**
- **Operational DBs (Postgres, Aurora, CosmosDB)**

Processing

- **Spark / Flink**
- **dbt**
- **Databricks / Synapse / EMR**

Governance

- **Collibra / Alation**
- **Unity Catalog / Purview / Lake Formation**
- **Great Expectations** (DQ)
- **OpenLineage / Marquez**

Analytics

- **Power BI / Tableau / Looker**
- **Trino / Presto**
- **Metrics layer (dbt metrics, AtScale)**

ML & GenAI

- **MLflow**
- **Feature Store**
- **SageMaker / Azure ML / Databricks ML**
- **Vector DB (Pinecone / OpenSearch / FAISS)**
- **GenAI Gateway (Bedrock / Azure OpenAI / Vertex)**
- **RAG Framework (LangChain / LlamaIndex)**

3️⃣ Application & Integration Stack

Frontend

- **React / Angular**
- **Mobile (React Native / Flutter)**

Backend

- **Java (Spring Boot)**
- **Python (FastAPI)**
- **Node.js**
- **.NET Core (if required)**

Integration

- **API Gateway (Apigee / Kong / AWS API GW / Azure APIM)**
- **Service Mesh (Istio / Linkerd)**
- **Event Bus (Kafka / EventBridge)**

Contracts

- **OpenAPI / AsyncAPI**
- **Schema Registry**

4️⃣ Infrastructure & Platform Stack

Cloud & Compute

- **AWS / Azure (primary)**
- **Kubernetes (EKS / AKS)**
- **Serverless (Lambda / Functions)**

Infrastructure as Code

- **Terraform**
- **Helm**

- **ArgoCD / Flux (GitOps)**

Networking

- **Private endpoints**
- **Zero Trust network access (Zscaler / Cloudflare)**
- **Load balancers**

Observability

- **Prometheus / Grafana**
- **OpenTelemetry**
- **ELK / OpenSearch**
- **Datadog / New Relic**

5️⃣ Security Stack (ZTA + HIPAA)

Identity

- **Azure AD / Okta / Ping**
- **OAuth2 / OIDC**
- **PAM (CyberArk)**

Secrets & Keys

- **HashiCorp Vault**
- **AWS KMS / Azure Key Vault**

Threat Detection

- **SIEM (Splunk / Sentinel)**
- **EDR/XDR (CrowdStrike)**
- **SOAR (Palo Alto Cortex)**

Security Testing

- **Snyk / Checkmarx / Veracode**
- **Trivy**

- **OWASP ZAP**

6️⃣ CI/CD & DevSecOps Stack

- **GitHub Actions / GitLab CI / Azure DevOps**
- **ArgoCD**
- **SonarQube**
- **SAST/DAST/SCA**
- **OPA / Sentinel integration**
- **Container scanning**
- **Policy-as-code gates**

7️⃣ Program & Portfolio Execution Stack

Planning

- **Jira / Jira Align**
- **Azure DevOps Boards**
- **SAFe tools (Rally)**

Portfolio

- **LeanIX / Planview**
- **ServiceNow PPM**

Reporting

- **Power BI / Tableau**
- **OKR tools (WorkBoard / Gtmhub)**

Risk & Compliance

- **ServiceNow GRC**
- **AuditBoard**
- **Confluence evidence repo**

8️⃣ Self-Service Enablement Stack

- **Backstage (Developer Portal)**
- **Golden path templates**
- **Terraform modules**
- **Pipeline templates**
- **OPA policies**
- **Service catalog**
- **API catalog**

9️⃣ Why This Stack Works

- Scales across products
- Enables self-service safely
- Embeds compliance
- Supports AI and GenAI
- Vendor-neutral
- Audit-ready
- Reduces cognitive load for teams
- Makes governance invisible

🎯 Interview-ready one-liner

“This stack enables self-service with guardrails — architecture defines the rules, platforms provide the tools, pipelines enforce compliance, and teams deliver independently at scale.”

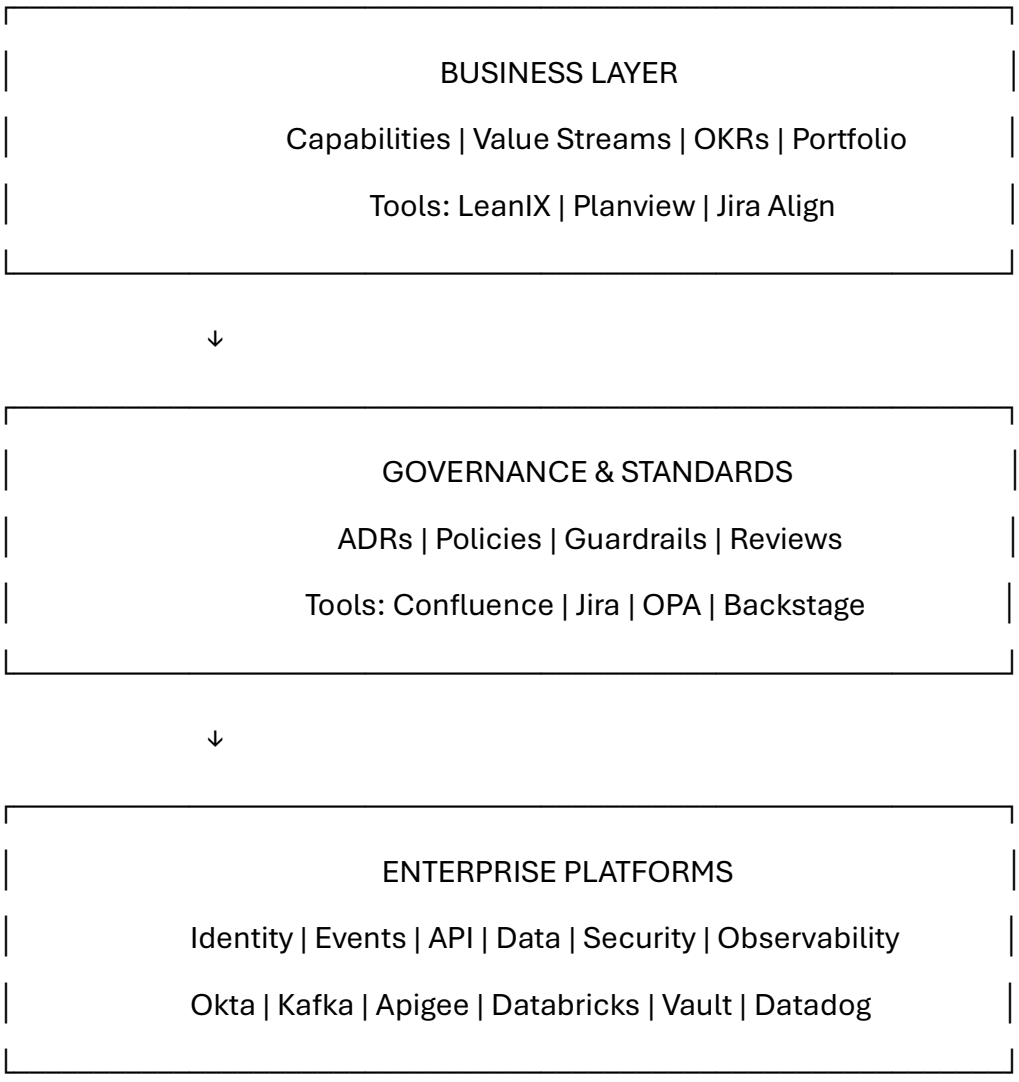
---

1. **Architecture diagrams with technology stack mapping** (clear, layered, copyable)
2. **Platform onboarding guide (step-by-step for teams)**
3. **Reference repository structures (ready to create in GitHub)**

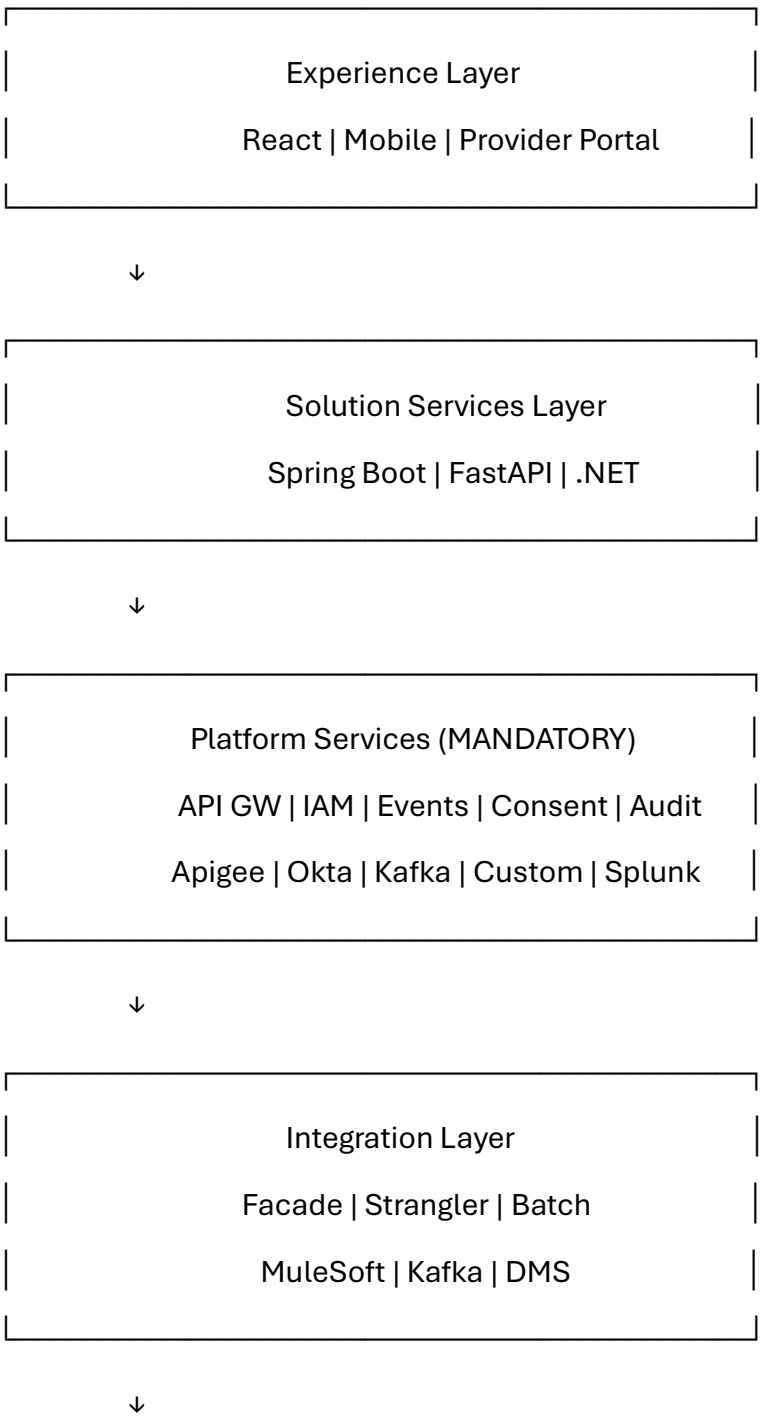
All aligned to **healthcare, HIPAA, ZTA, self-service, multi-product scale**.

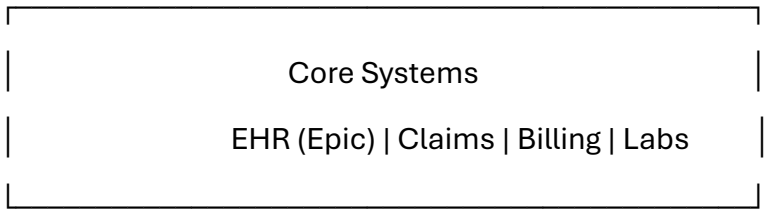
1️⃣ Architecture Diagrams with Stack Mapping

1.1 Enterprise Architecture (Stack-Mapped)

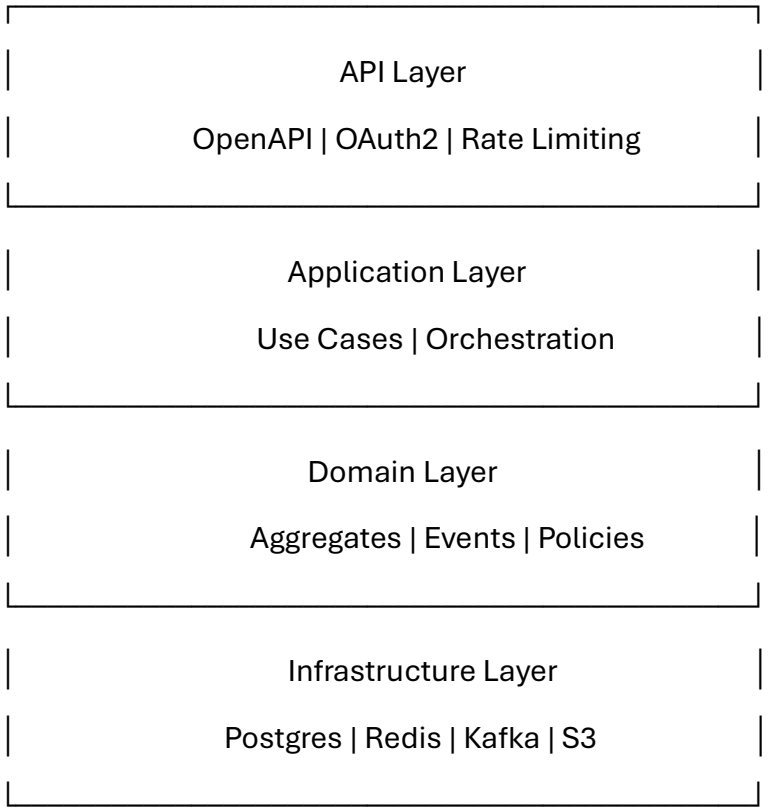


1.2 Solution Architecture (Program-Level, Stack-Mapped)





1.3 Application Architecture (Service-Level, Stack-Mapped)



CI/CD: GitHub Actions | ArgoCD

Security: Vault | Snyk | Trivy

Observability: OpenTelemetry | Datadog

1.4 Data + AI Architecture (Stack-Mapped)

Sources → Ingestion → Lakehouse → Semantic → Analytics/AI

EHR → Kafka → Delta Lake → FHIR Models → Power BI

Apps → CDC → Iceberg → Metrics → MLflow

IoT → API → Feature → Features → GenAI (RAG)

Stack

- Ingestion: Kafka, DMS, Fivetran
- Lakehouse: Databricks, S3, ADLS
- Governance: Collibra, Purview
- ML: MLflow, SageMaker
- GenAI: Bedrock / Azure OpenAI + RAG

1.5 Security (ZTA) Architecture

User → IdP → PDP → PEP → App/Data/AI

| | |

Okta OPA Envoy

- Identity: Okta / Azure AD
- Policy: OPA
- Enforcement: API GW, Service Mesh
- Audit: Splunk / Sentinel

2️⃣ Platform Onboarding Guide (Team Self-Service)

This is **exactly what new teams follow** when joining the platform.

Platform Onboarding – 30/60/90 Day Guide

🟢 Day 1–30: Understand & Access

Mandatory

- Read enterprise reference architecture
- Complete security training (HIPAA, PHI)
- Get access to:
    - Backstage portal
    - GitHub org
    - Cloud subscription
    - Jira project
- Review golden path examples

Output

- Team registered in platform catalog
- Owners assigned
- First ADR created

🟡 Day 31–60: Build & Integrate

Self-Service Actions

- Generate service from template
- Provision infra using Terraform module
- Create CI/CD pipeline
- Register API

- Register schema
- Enable audit logging
- Enable monitoring

Output

- First service deployed
- First data product registered
- Platform services integrated

🔵 Day 61–90: Scale & Optimize

Required

- Add contract tests
- Enable cost monitoring
- Add DQ rules
- Enable DR
- Review architecture health metrics
- Close onboarding checklist

Output

- Production-ready service
- Audit-ready
- Fully governed
- Zero manual approvals

3️⃣ Reference Repositories (GitHub-Ready)

3.1 Platform Reference Repos (Owned by Platform Team)

platform-standards/

```
├── architecture/
├── adr-template/
├── diagrams/
├── policies/
└── checklists/
```

platform-terraform-modules/

```
├── network/
├── eks/
├── databases/
├── observability/
└── security/
```

platform-policies-opa/

```
├── security/
├── data/
├── architecture/
├── ai/
└── infra/
```

platform-golden-paths/

├── backend-service/

├── data-product/

├── api-service/

├── ml-pipeline/

└── genai-app/

3.2 Team Reference Repos (Generated via Template)

product-service-template/

├── src/

├── api/

├── infra/

├── ci/

├── security/

├── observability/

├── runbook/

└── adr/

data-product-template/

├── ingestion/

├── schema/

├── quality/

├── lineage/

├── access/

└── docs/

3.3 CI/CD Reference Repo

pipeline-templates/

├── build.yml

├── test.yml

├── security.yml

├── deploy.yml

├── policy-check.yml

└── evidence.yml

3.4 GenAI Reference Repo

genai-governed-template/

├── prompts/

├── rag/

├── policies/

├── validation/

├── audit/

└── approval/

🧠 How This All Fits Together

Problem	Solved By
Slow onboarding	Golden paths
Inconsistent architecture	Reference repos
Manual governance	OPA + pipelines

Problem	Solved By
Security friction	ZTA automation
Audit pain	Evidence as code
Platform bypass	Make platform easiest

🎯 Executive / Interview Summary

“We scale architecture by combining clear diagrams, self-service onboarding, and reference repositories. Teams start fast, comply automatically, and never need permission to do the right thing.”
