Elevens Bank

Next-Gen Cloud-Native Microservices Architecture

A reference implementation for scalable, fault-tolerant financial applications using Java 21 and the Spring Ecosystem.

ARCHITECTURE OVERVIEW & TECHNICAL DEEP DIVE

The Blueprint for Modern Distributed Banking

Elevens Bank solves the complexity of distributed systems by leveraging industry-standard patterns for consistency, security, and observability.

Cloud-Native Foundation

Built on `Java 21`, `Spring Boot 3.3`, and `Spring Cloud 2023`. Containerized with `Docker`.

Event-Driven Core

Asynchronous service decoupling using `Apache Kafka` for high throughput.

Resilience First

Circuit Breakers, Rate Limiting, and Bulkheads using `Resilience4j`.

Advanced Consistency

Managing distributed transactions via the SAGA Choreography pattern.

Deep Observability

Full distributed tracing with `Micrometer`, `OpenTelemetry`, and `Zipkin`.

Responsive Single-Page Application Interface

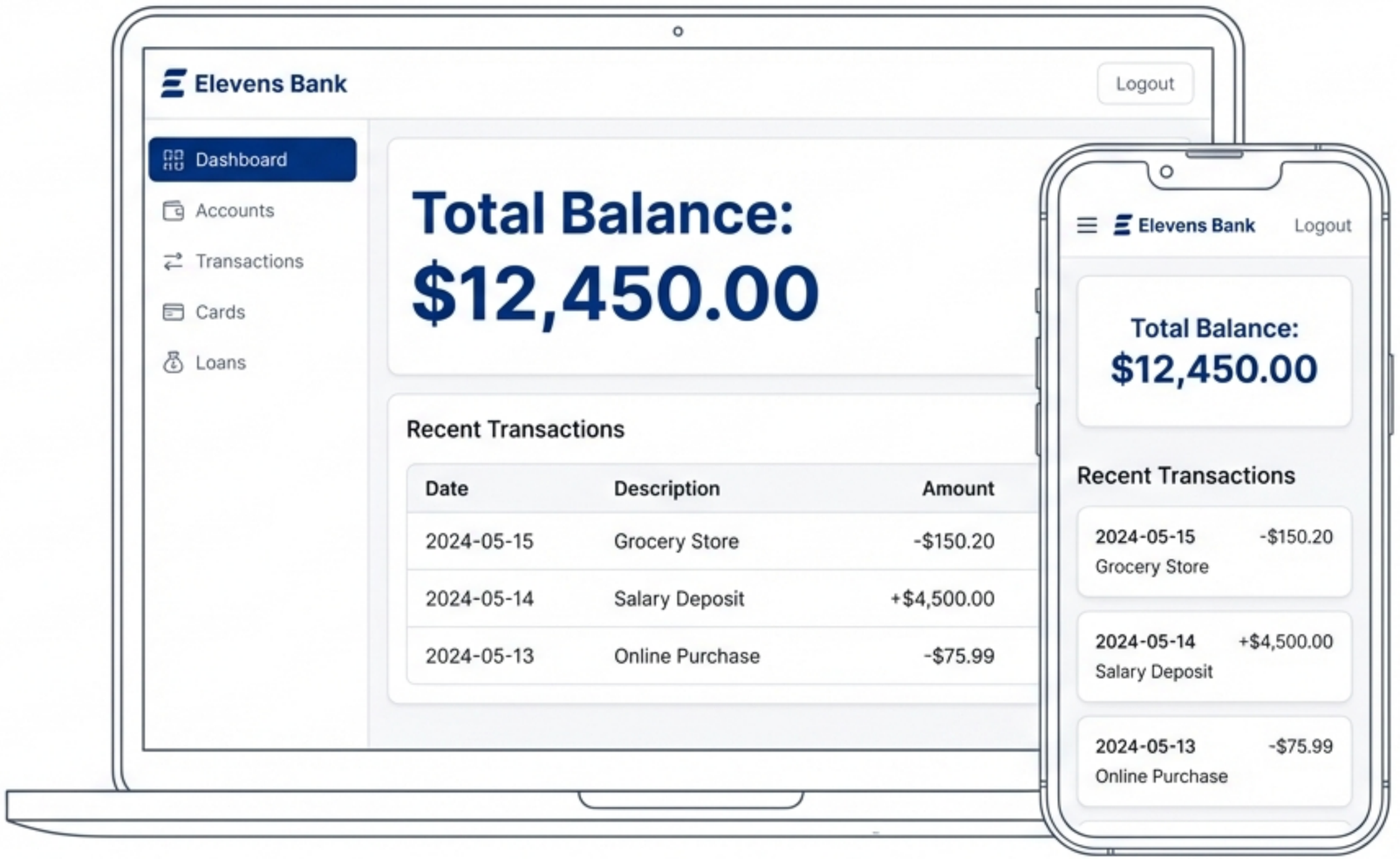
A modern React-based dashboard consuming the microservices mesh.

Tech Specs

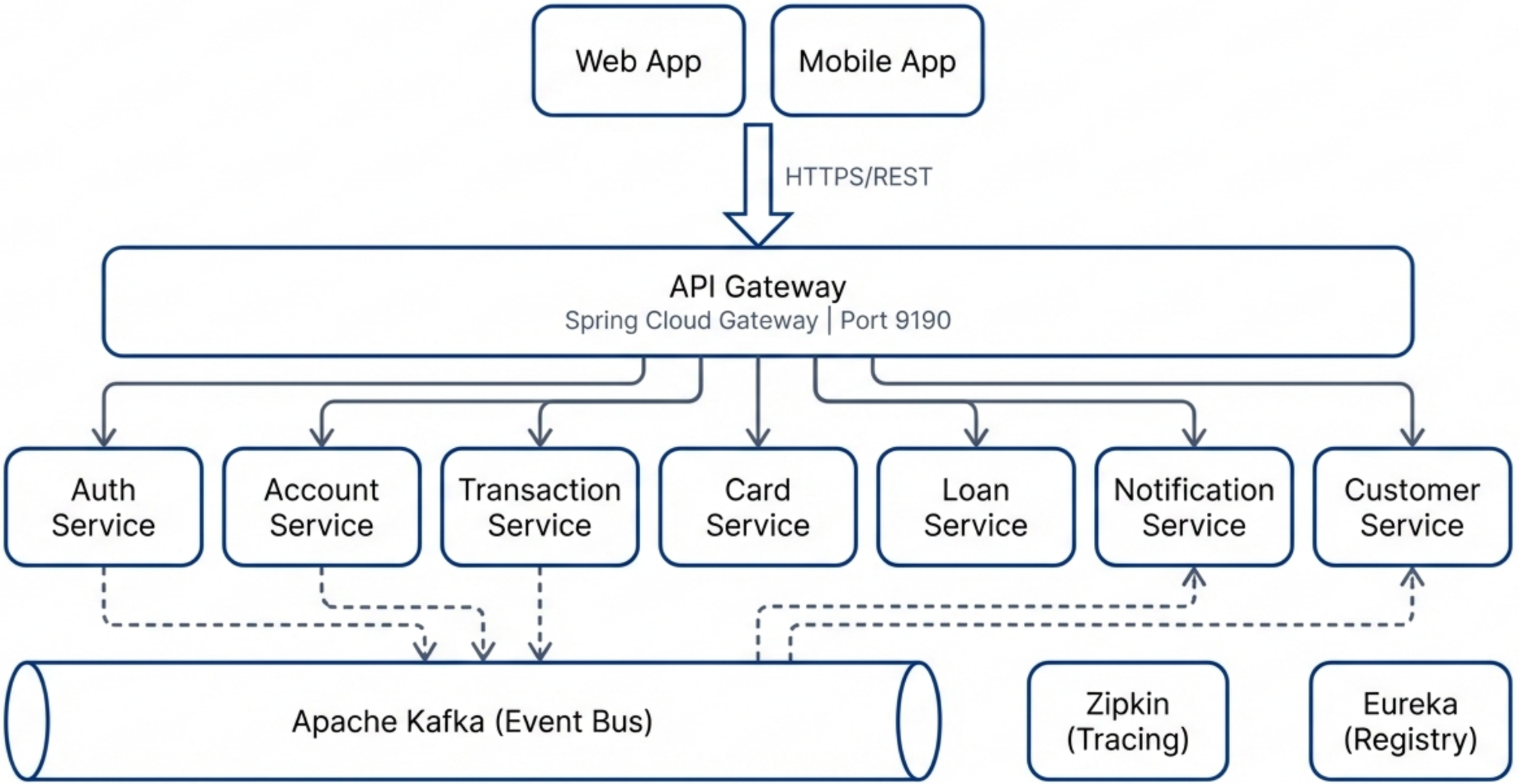
- Framework: `React (v16+)`
- Styling: `TailwindCSS`
- State/Routing: `React Context`, `React Router`
- HTTP Client: `Axios` (with JWT interceptors)

Key Features

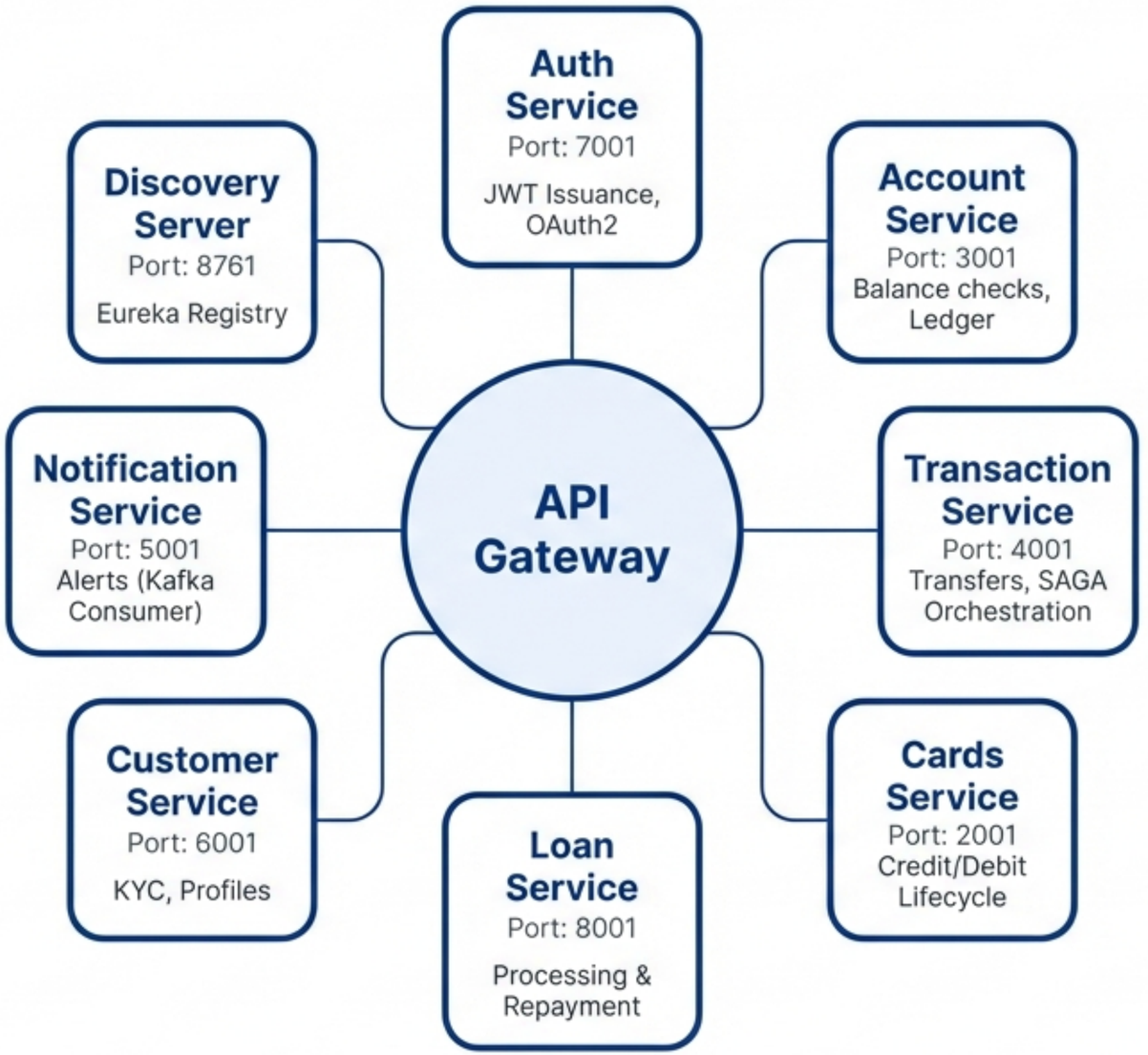
- Real-time account updates
- Secure authentication flows
- Loan & Card management modules



High-Level System Topology (C4 View)

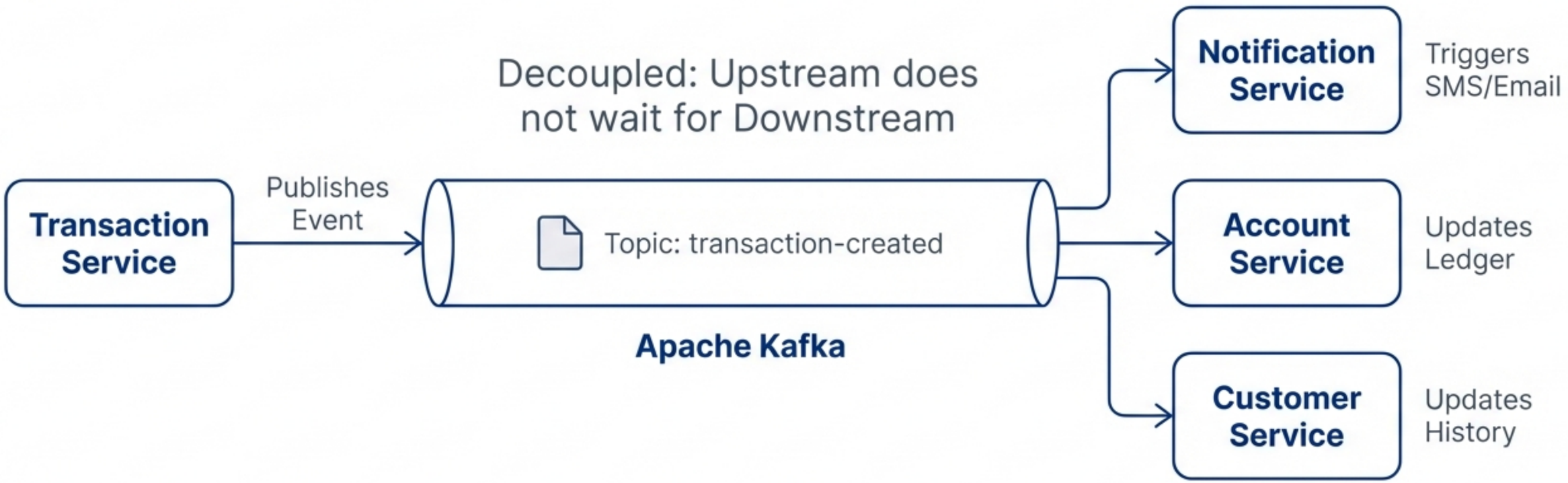


The Microservices Landscape



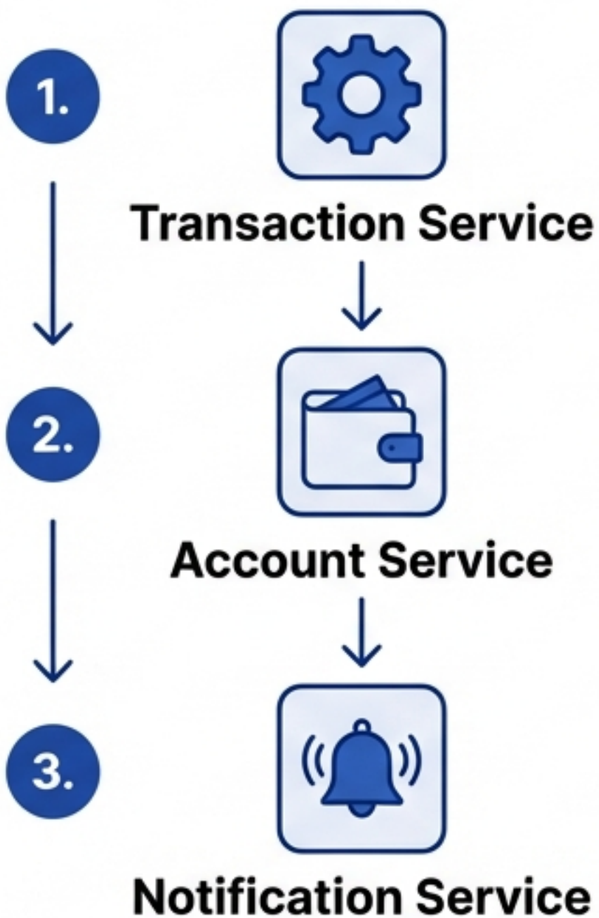
Event-Driven & Asynchronous Communication

Decoupling services using Apache Kafka for high scalability.



Distributed Consistency via SAGA

Solving the “dual-write” problem with Choreography.



Initiates Transfer & Publishes “Initiated” Event

Consumes Event -> Deducts Funds -> Publishes “Success” or “Failure" Event

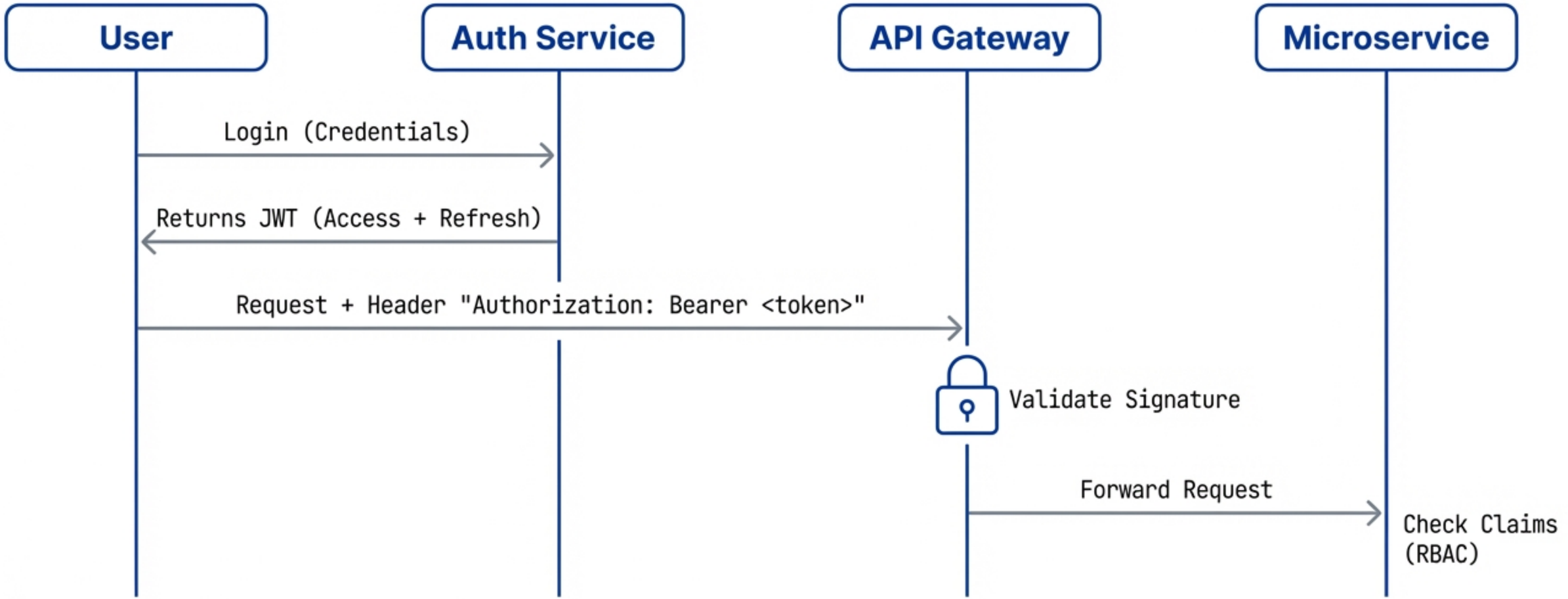
Consumes Result -> Alerts User via Email/SMS

Mechanism

- No central coordinator.
- Services react to events (Choreography).
- Ensures eventual consistency across distributed databases.

Security Architecture & Access Control

Stateless Authentication with OAuth2 and JWT.



Tech: Spring Security 6, OAuth2 Resource Server.

Engineering Resilience

Preventing cascading failures with Resilience4j.

Circuit Breakers

Stops requests to failing services to prevent resource exhaustion.

Rate Limiting

Throttles request frequency to protect against traffic spikes.

Retries

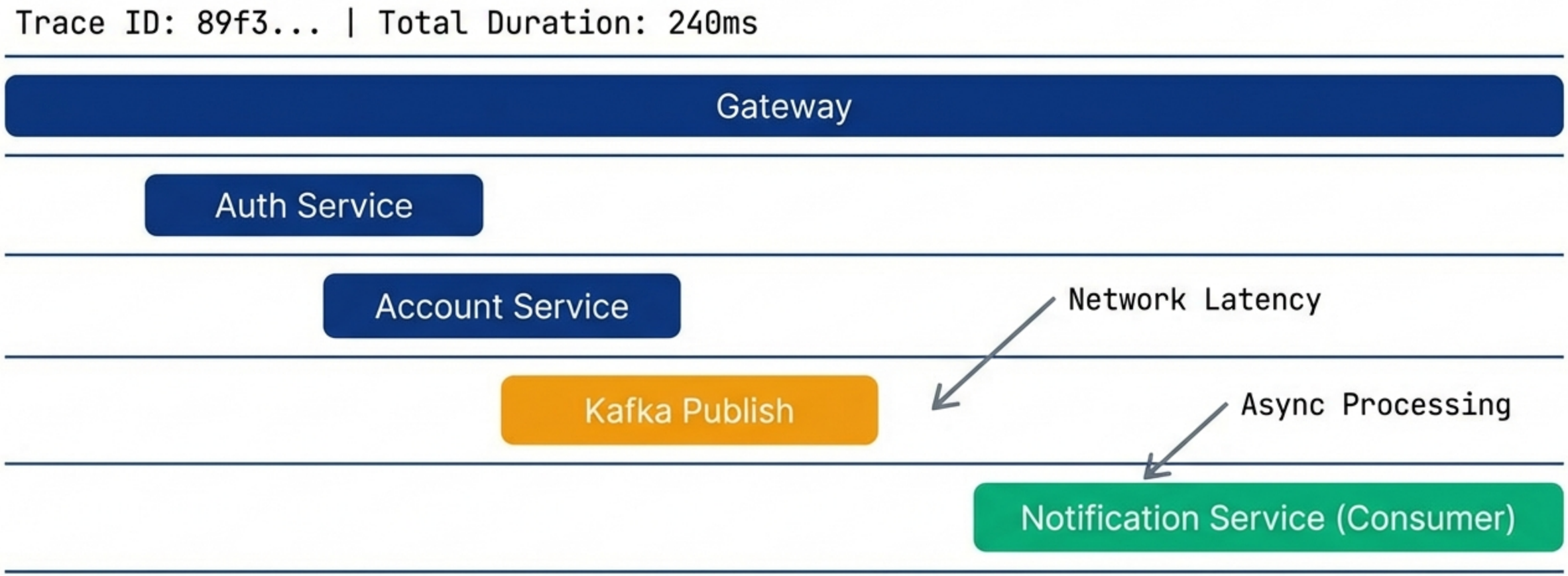
Automatically retries transient network failures.

Bulkheads

Isolates resources so one component failure doesn't crash the system.

Observability & Distributed Tracing

Full visibility via Micrometer, OpenTelemetry, and Zipkin.



Zipkin running on port 9411 collects traces across all service boundaries.

Comprehensive Technology Stack

Categories	Technologies
Core Framework	Java 21 LTS, Spring Boot 3.3, Spring Cloud 2023
Communication	REST, GraphQL, Apache Kafka, OpenFeign
Data Persistence	MySQL (Transactional), MongoDB (Logs)
Security	Spring Security 6, OAuth2, JWT
Resilience	Resilience4j, Spring Cloud Circuit Breaker
Observability	Micrometer, OpenTelemetry, Zipkin
Infrastructure	Docker, Docker Compose, Netflix Eureka

Developer Experience & Deployment

Prerequisites

- Java 21
- Docker & Docker Compose
- Maven 3.9+

API-First Design

- Contract-first development.
- OpenAPI/Swagger documentation available for every service.

```
$ git clone https://github.com/shrikar-dev/elevens-bank.git
$ cd elevens-bank
$ docker-compose up -d
[+] Running 8/8
  ✓ Container kafka       Running
  ✓ Container zookeeper   Running
  ✓ Container zipkin      Running
  ✓ Container mysqldb     Running
...
```

Single command infrastructure provisioning.

Architecture Summary

Scalable

Decoupled services allow independent scaling of high-load domains.

Secure

Industry-standard `OAuth2/JWT` implementation with granular `RBAC`.

Resilient

Fault-tolerant design handles outages gracefully without data loss.

Transparent

Complete visibility into every transaction via distributed tracing.

Modern

Built on the latest LTS `Java 21` and `Spring Boot 3.3` versions.

Explore the Code

Get hands-on with the reference implementation.

Author: Shrikar

Senior Java Developer

Backend Repository

`github.com/shrikar-dev/elevens-bank`

Frontend Repository

`github.com/ShrikarMukesh/elevens-bank-ui`

Clone. Build. Deploy.