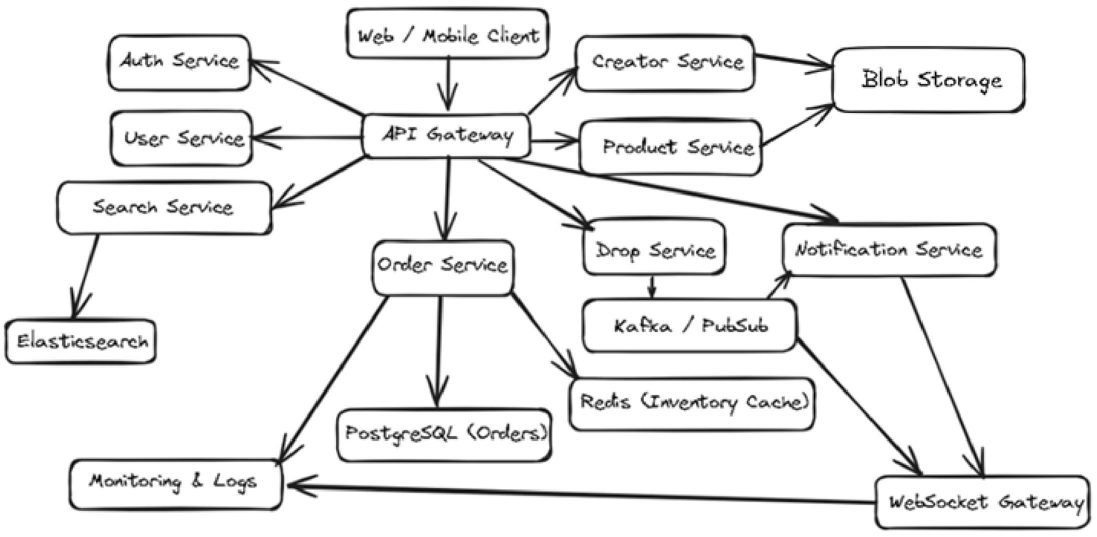
LiveDrop System Design Report

This document outlines the system design for "LiveDrop" - a flash-sale platform that enables creators to launch limited-inventory product drops with real-time notifications and robust ordering infrastructure.

Author: Charbel Chalouhy

Architecture Diagram



Key Features

- Real-time drop notifications via Kafka + WebSocket
- Atomic stock handling with Redis (no overselling)
- Follower system supporting celebrity-scale users
- Full-text product search (Elasticsearch)
- Secure, idempotent ordering API with retries
- Low-latency APIs and paginated responses

    - Horizontal scaling with stateless services

Public API Overview

- POST /creators/:id/follow - Follow a creator

- DELETE /creators/:id/follow - Unfollow a creator

- GET /creators/:id/followers - List followers (paginated)

- GET /products - Browse/search products

- GET /drops/:id - View a drop

- POST /orders - Place an order (with idempotency key)

- GET /orders/:id - Retrieve order status

Data Models Overview

Relational DB: PostgreSQL

- Tables: users, creators, products, drops, orders, follows

In-memory DB: Redis

- Used for atomic stock locking and caching hot data

Search Engine: Elasticsearch

- Indexes product names and descriptions

Object Storage: Blob Storage

- Hosts product images and creator media

Design Tradeoffs

- Redis was chosen for high-performance stock operations via Lua scripting.

- PostgreSQL handles transactional order data and relationships.

- Kafka decouples services and enables real-time event streaming.

- WebSocket gateway enables real-time client updates.
- Elasticsearch powers search while reducing DB load.
- Idempotency keys prevent duplicate orders under retries.