ShortlyX

Short-Form Video Platform — Transcoding Pipeline & Personalized Feed

Scalable demo-ready architecture designed to support reliable ingestion, transcoding, distribution, and personalized content delivery of short videos

SUKHRAJ SINGH

Problem Statement — Short-Form Video Platform

Ingest-to-feed challenges: scalability, cost, latency, personalization, and abuse mitigation

01  **Reliable transcoding and global CDN delivery for smooth playback**

02  **Handle heavy write-heavy ingest workloads from concurrent uploads**

03  **Provide ranked, personalized feed to drive engagement with low tail latency**

04  **Control storage costs while retaining availability and retention needs**

05  **Minimize tail latency to keep feed responsiveness and reduce user friction**

Goals & Success Metrics

Measurable KPIs for ingest, availability, cost efficiency, and engagement

Ingest capacity: Target peak ingest **QPS** to handle peak upload demand reliably

Latency: Minimize end-to-end latency from upload to playback for a seamless user experience

Availability: Ensure **99.9%** system availability to support continuous service

Cost efficiency: Optimize **cost per GB stored monthly** balancing performance and expenses

Engagement: Measure user engagement via **watch time** and **CTR** to validate feed effectiveness

Core Features

End-to-end ingestion, transcoding, distribution, social and analytics capabilities

01

Ingestion: Upload APIs with resumable uploads for network resilience

02

Processing: Serverless + batch transcoding producing multiple playback profiles (HLS/DASH)

03

Distribution: CDN distribution with playback optimizations and adaptive streaming

04

Social: Likes, comments, moderation tools for community control

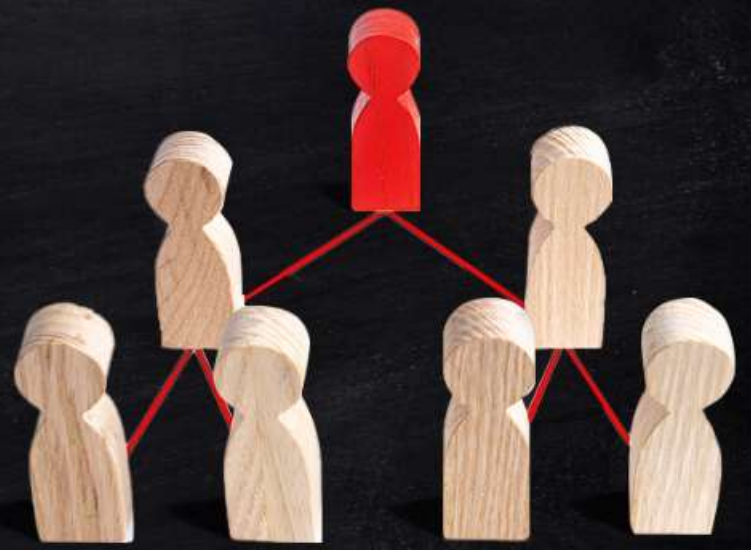
05

Analytics: Event logs and behavioral insights for monitoring and product decisions

Non-Functional Requirements: Scaling, Latency, Cost, Resilience

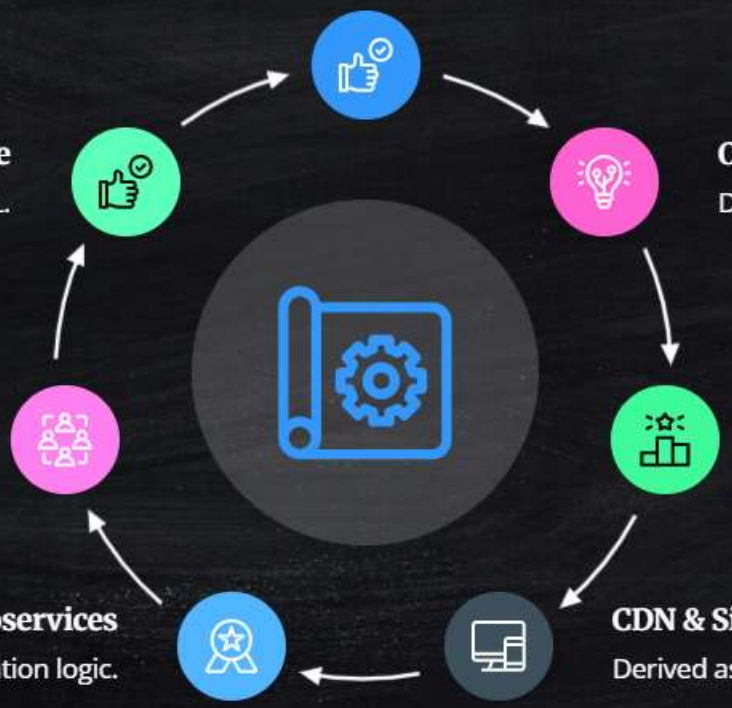
Operational constraints for a write-heavy short-form video platform with personalized feed

- Scalability: optimized for write-heavy ingest workloads; autoscale ingestion and transcoding horizontally to handle bursty uploads
- Latency: maintain low tail latency for feed generation (95th/99th percentile) to ensure responsive user experience
- Cost: efficient cold storage tiering and cost-aware autoscaling to minimize spend during low traffic
- Resilience: multi-region deployment for high availability and disaster recovery with cross-region failover
- Operational: observability, SLOs for feed latency and ingest throughput, and automated scaling policies tied to cost and performance



High-Level Architecture

End-to-end flow from client auth and ingest to personalized feed, CDN delivery, and analytics



Clients & API Gateway

Client auth, upload routing and request validation at edge.

Object Storage (Raw Video)

Durable raw video staging before processing.

Transcoding Pipeline

Serverless + batch workers produce multiple variants for playback.

CDN & Signed URLs

Derived assets distributed via CDN; signed URLs for secure playback.

Core Microservices

Feed ranking, social graph, comments, notifications handle application logic.

Redis Cache

Low-latency state and session access for UX-critical paths.

Analytics Pipeline

Event aggregation into columnar warehouse for reporting and ML.

Upload & Ingest Flow

Resumable upload → queued transcoding → derived assets → CDN invalidation and manifest for immediate playback

Client resumable upload

Clients upload chunks to object storage with resumable protocol for fault tolerance and retry.

Upload completion triggers enqueue

On final chunk, system enqueues a job to kick off the transcoding pipeline.

Transcoder processes variants

Transcoder generates required bitrate/resolution **variants** and captions/metadata as needed.

Write derived assets to storage

Transcoded variants and derived assets are written back to object storage with consistent keys.

CDN cache invalidation

Invalidate affected CDN paths to ensure edge caches serve latest assets after write.

Manifest file generation

Generate playback manifests (HLS/DASH) referencing new variants to enable immediate playback.

Transcoding Pipeline Design

Builder-stage pipeline with resilient error handling, thumbnails, and autoscaling

CDN Delivery & Signed URLs — Secure, Fast Media Delivery

How caching, signing, origin failover, and cache-control keep video delivery low-latency and protected

Edge caching with configurable TTLs to minimize latency while balancing freshness and performance

Signed short-lived URLs (tokenized, time-limited) to prevent unauthorized access and link abuse

Origin failover routes requests to healthy origins when CDN nodes or POPs fail to ensure availability

Cache-control policies (Cache-Control, ETag, Vary) to optimize bandwidth and reduce origin load

Combine TTLs + short-lived signed URLs + origin failover for **resilient, secure, low-latency** video delivery

Feed Ranking & Personalization

Signals, hybrid model choices, and A/B testing to optimize short-form feed

01 **Ranking signals:** recency, **watch completion rates**, social graph, user preferenc device/context

02 **Model options:** heuristic score + ML re-ranker for personalization; end-to-end M when scale permits

03 Re-ranker objectives: relevance, diversity, freshness, engagement lift

04 **A/B testing:** online experiments for ranking variants, metric set: watch completio CTR, session length, retention

05 Safety & business constraints enforced post-rank (policy filters, revenue rules)

06 Monitoring: drift detection, offline holdouts, daily experiment dashboards

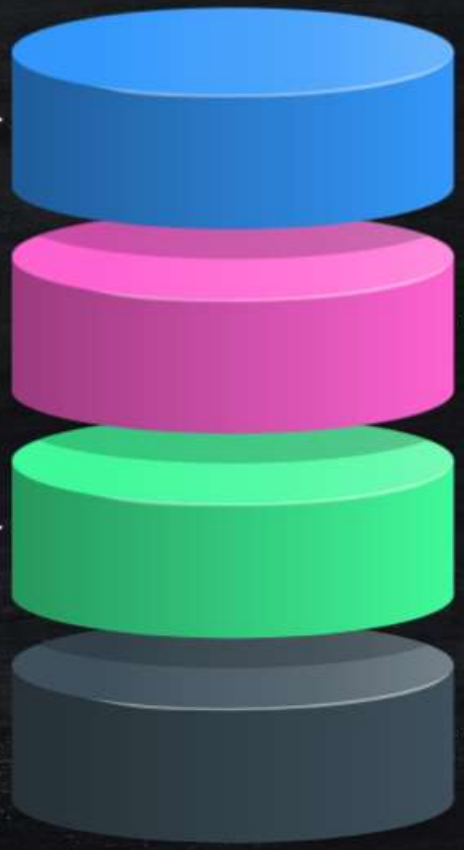
Data Stores & Schema Overview

Roles, retention and tiering across storage layers for short-form video platform

Object Storage (raw & derived media)

01

Stores original uploads, transcoded assets; cost-effective tiering (hot/cold/archival) for retention and cost control.



Relational DB (profiles & social graph)

02

User profiles and social graph ensuring consistency, transactional integrity, and low-latency queries for relationships.

Redis Cache (hot state)

03

Caches trending content, session state, and hot lookups for sub-ms access and load reduction on primary stores.

Columnar Warehouse (analytics)

04

Batch & streaming ingestion for analytics; supports retention policies, tiering, and large-scale query workloads.

Observability & Analytics

Real-time event capture, SLOs, tracing, and analytics pipelines for the transcoding feed

Event bus captures real-time and batch streams for full observability

Metrics & dashboards monitor critical **SLOs**: transcoding latency, system availability

Tracing instruments tail latency to pinpoint bottlenecks across pipeline stages

Data pipelines feed analytics warehouse for user behavior analysis and reporting

Combine real-time metrics + batch analytics for operational alerts and product insights

Security & Moderation

Layered defenses for safe uploads, controlled access, and regulatory compliance

Automated scanning pipeline: **auto-scanners** + curated human reviewers detect inappropriate uploads

Abuse mitigation via **rate limiting** to enforce fair usage and throttle anomalous traffic

Access protection using **signed URLs** for content and **role-based access controls** for ops

Privacy & retention: policies align with relevant regulations; enforce data lifecycle controls

Moderation flow: ingest → auto-scan → flag → human review → publish/quarantine

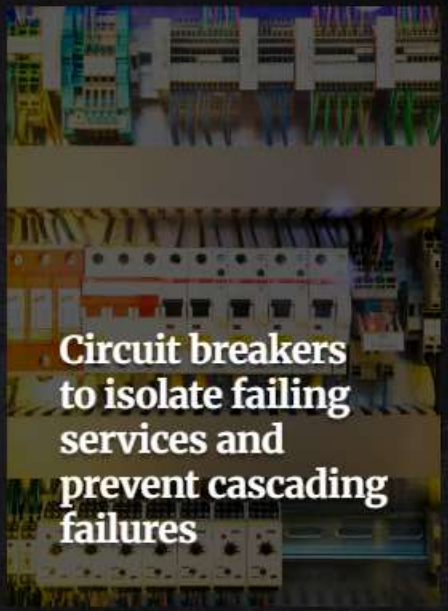
Operational focus: logs, audit trails, and alerts to support compliance and incident response

Reliability & Scaling

Patterns to keep video feed available, performant, and resilient at scale

Multi-region replication for fault tolerance and disaster recovery

Graceful degradation: serve lower-resolution variants when full-quality assets unavailable



Circuit breakers to isolate failing services and prevent cascading failures

Retries with exponential backoff and jitter to reduce load spikes

Autoscale critical components based on traffic and pipeline lag metrics

Cost & Operational Considerations

Practical levers to reduce spend and maintain reliable transcoding & storage

- Use **storage tiering** (hot/warm/cold) to align cost with access patterns
- Run transcoding on **spot/ephemeral workers** for variable demand and lower compute cost
- Monitor **cost per play** and per-transcode to surface inefficiencies
- Enforce **data retention policies** to control long-term storage and meet compliance
- Automate **tier transitions** and spot bidding with SLO-aware rules

Roadmap & MVP Plan

Phased rollout of core upload, transcoding, CDN delivery, and personalized feed

MVP scope: core upload, transcoding, CDN delivery, personalized feed

Phases: initial demo → regional beta → full global launch

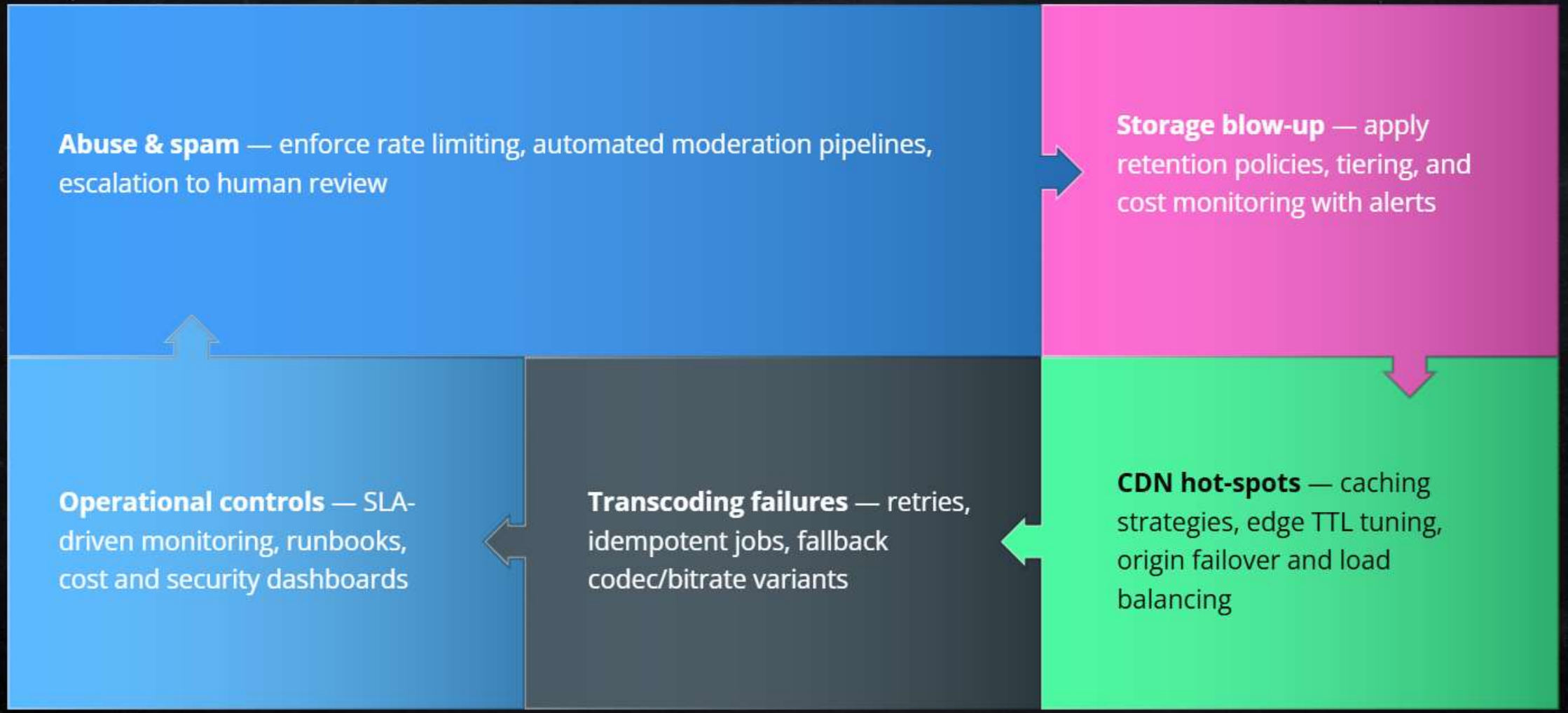
Testing strategy: integration tests and canary deployments

Validation metrics: ingest throughput, latency, user engagement

Milestones: demo-ready build, regional SLA verification, global scale validation

Risks & Mitigations for Video Platform

Categorized risks with targeted operational and architectural controls



Appendix — API & Design Patterns

Resumable upload sample, event schema example, and core architecture patterns

Resumable Upload API: POST /uploads to create session; PATCH /uploads/{id} for chunked resumable upload; GET /uploads/{id} to check status

Event Schema (analytics): event_type, timestamp, user_id, video_id, stage, metadata (latency, error_code)

Pipeline/Builder: modular **transcoding stages** (ingest → decode → transcode → package → store)

Strategy: **ranking** strategies pluggable for personalized feed (engagement, recency, diversity)

Observer: event listeners for telemetry, retries, and downstream notifications

Facade: simplified interface for subsystems (storage, CDN, transcode, analytics)

Thank You

Questions and discussion welcome !!