

MXFlix Technical Product Dossier

Enterprise OTT Streaming Platform – Comprehensive Technical Evaluation

A rigorous technical analysis of the MXFlix OTT streaming platform covering architecture, database design, API surface, security, video delivery, monetization, mobile clients, and operational characteristics. All data derived from live schema and route analysis.

VERSION 1.0

TECHNICAL DUE DILIGENCE

ARCHITECTURE REVIEW

Executive Summary

This document provides an exhaustive technical analysis of the MXFlix OTT streaming platform. MXFlix is a feature-complete, enterprise-grade streaming solution built on Laravel 12 with PHP 8.2+, delivering a modular monolith architecture that supports SVOD, TVOD, and AVOD revenue models concurrently. The platform surfaces 1,135 routes across four distinct access tiers, maintains 94 database tables with 1,229 total columns, and ships Flutter-based mobile clients for iOS, Android, Web, and Desktop from a single codebase.

The evaluation covers every major technical domain: core architecture and module boundaries, complete database schema documentation by domain, API surface and versioning strategy, six-method authentication system, video delivery mechanisms, monetization engine, background processing, storage backends, notification system, analytics, mobile architecture, security posture, white-label configurability, performance characteristics, third-party integrations, operational requirements, and a structured risk assessment. Each section derives findings directly from live schema and route analysis without inference or extrapolation.

Architects and platform engineers performing technical due diligence will find this dossier structured to support rapid domain-specific review. Database schema sections enumerate every column with type, nullability, and key classification. API sections list all routes with HTTP method and named route. Risk sections quantify likelihood and impact for operational planning. The document is intended as the authoritative technical reference for procurement, integration planning, and deployment scoping decisions.

94

Database Tables
Across 9 functional domains

1229

Total Columns
Full schema coverage

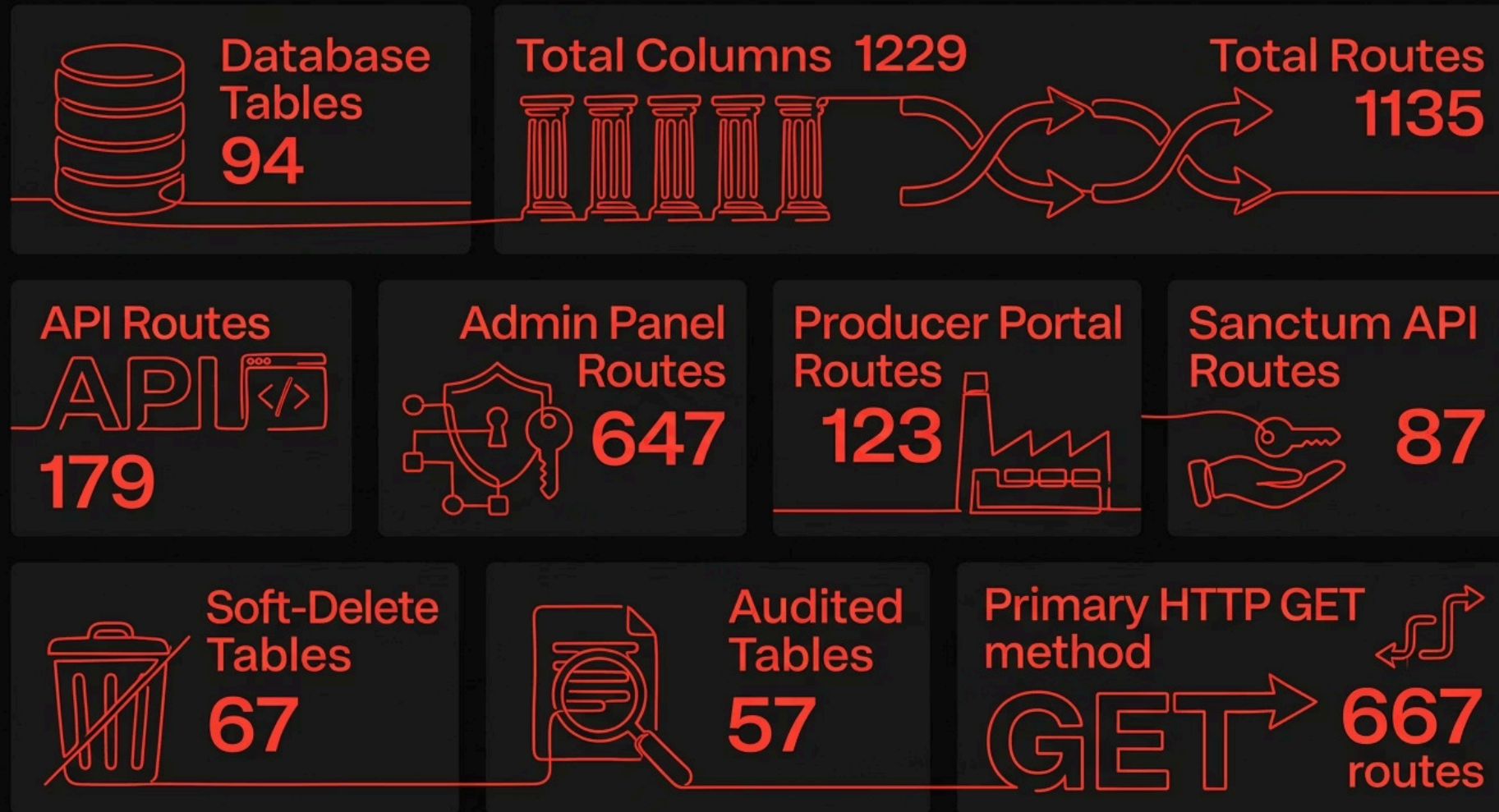
1135

Total Routes
Across 4 access tiers

179

REST API Routes
Sanctum-authenticated

Platform Metrics at a Glance



These metrics derive from live schema and route analysis. The distribution of routes — 647 admin, 123 producer, 179 REST API — reflects a platform built equally for operational control, content partner workflows, and mobile client consumption. The 67 soft-delete tables and 57 audited tables underscore a design philosophy that prioritizes data recoverability and compliance traceability across all critical entities.

1. Architecture Overview

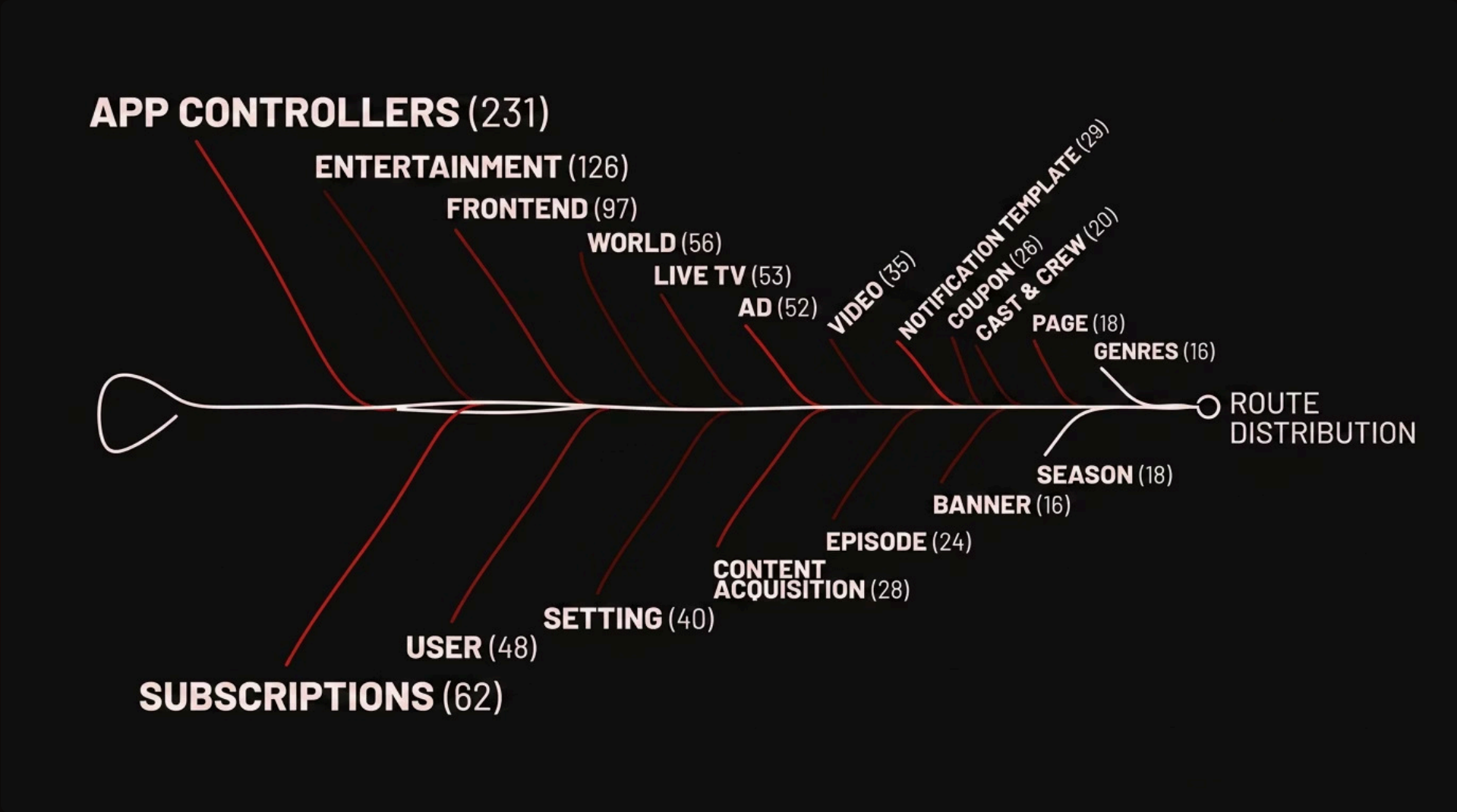
MXFlix is a modular, enterprise-grade OTT streaming platform built on Laravel 12 with PHP 8.2+. The architecture follows a modular monolith pattern where business domains are encapsulated as independent modules, each carrying their own controllers, models, views, and routes — providing clear domain boundaries without the operational overhead of a distributed microservices topology.

Core Technology Stack

Layer	Technology	Purpose
Backend Framework	Laravel 12 (PHP 8.2+)	REST API, Queue Workers, Authentication
Database	MySQL 8.x	Relational data persistence
Cache/Session	Redis	Session management, caching, queue driver
Mobile	Flutter (Dart)	iOS, Android, Web, Desktop clients
Video Player	Video.js	Web-based adaptive streaming
Queue	Laravel Horizon/Redis	Background job processing
Auth	Laravel Sanctum	API token authentication
Permissions	Spatie Laravel Permission	RBAC with roles and permissions
Media	Spatie Media Library	File management and conversions
Audit	Spatie Laravel Activity Log	Full audit trail

Modular Domain Architecture

The platform is organized into 20+ distinct business modules, each with dedicated controllers, routes, and middleware. This modular monolith approach enables independent feature toggling, per-module permission control, and clearly bounded maintenance responsibilities without introducing distributed systems complexity.



Key Architectural Patterns

Modular Monolith

Business domains are isolated modules with clear boundaries, enabling independent feature toggling and maintenance without microservices overhead.

Middleware Pipeline

Every request passes through layered middleware for installation checks, authentication, authorization (admin/producer/user), and rate limiting.

Soft-Delete Pattern

67 tables implement soft-deletes via `deleted_at` timestamps, enabling data recovery and audit compliance across all critical entities.

Audit Trail

57 tables track `created_by`, `updated_by`, `deleted_by` columns for full change attribution and regulatory compliance.

Event-Driven

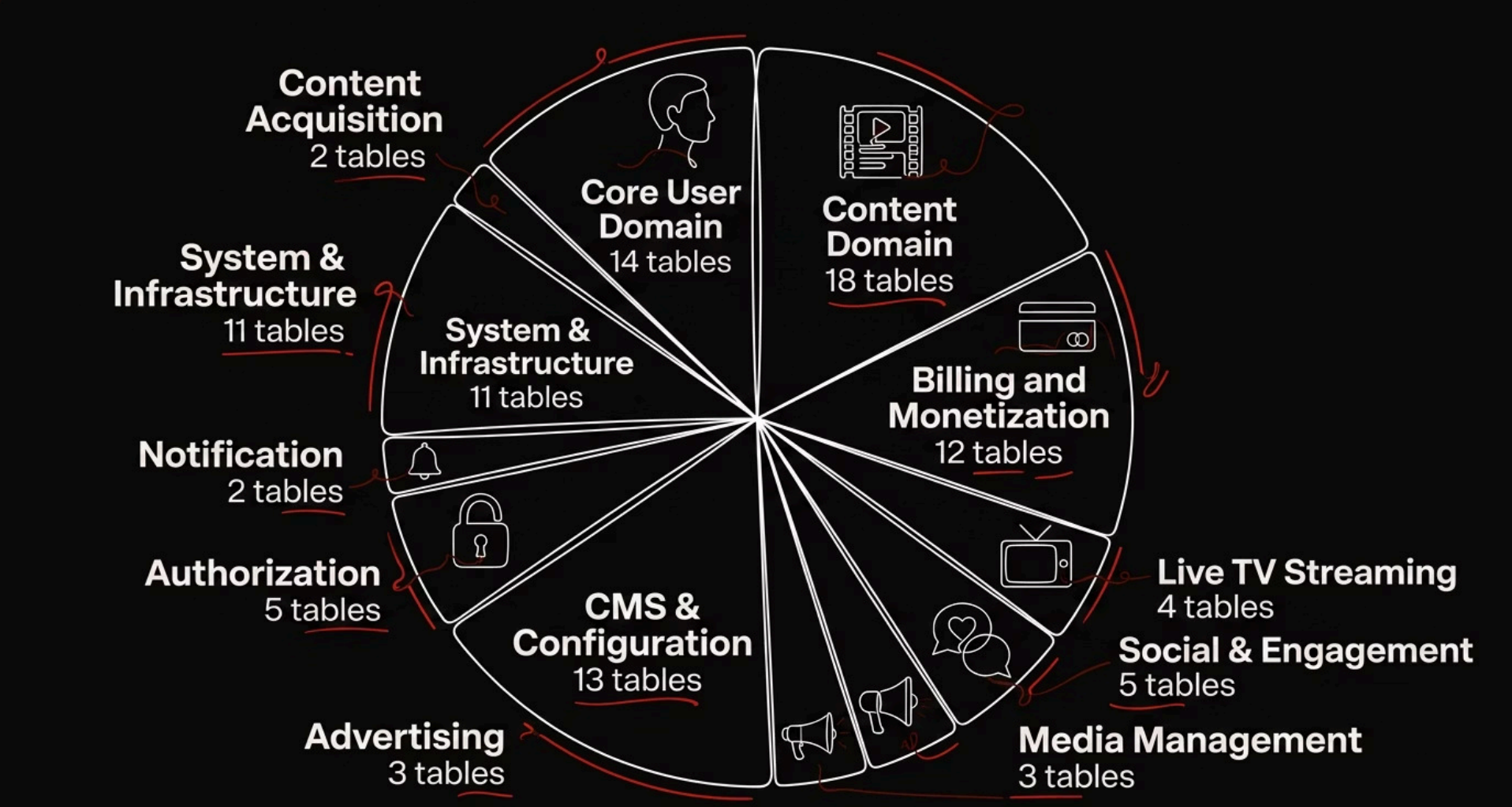
Laravel Notifications and event listeners power async workflows for email, push, and SMS delivery across all user-facing communication channels.

Route Access Tiers

Access Tier	Route Count	Authentication	Description
Admin Panel	647	Session + Role	Full CMS with role-based access
Producer Portal	123	Session + Producer	Content submission and management
REST API	179	Sanctum Token	Mobile and third-party integrations
Throttled API	48	Sanctum + Rate Limit	Protected endpoints with rate limiting

2. Database Architecture

MXFlix maintains 94 database tables with 1,229 total columns on MySQL 8.x. The schema follows Laravel conventions with Eloquent ORM abstraction and is organized into 9 functional domains, each with its own set of tables governing a specific business capability. All primary keys use bigint unsigned for large-scale scalability, and every table carries standard created_at/updated_at timestamps.



Schema Design Patterns

Pattern	Implementation	Tables Affected
Soft Deletes	deleted_at nullable timestamp	67 tables
Audit Columns	created_by/updated_by/deleted_by	57 tables
Auto-increment PK	id bigint unsigned	All tables
Polymorphic Relations	model_type/model_id	media, notifications, addresses
UUID Tracking	batch_uuid	activity_log
Enum Columns	enum(...)	videos (approval_status), currencies
Decimal Precision	decimal(10,4)	videos (revenue_per_view)

2.1 Core User Domain (14 Tables)

The user domain governs identity, session management, device tracking, and engagement scaffolding. The central users table carries 32 columns and is the pivot entity for the entire platform — linking subscriptions, profiles, devices, watch history, and payment details. User identity is extended through multi-profile support, OAuth provider linkage, and dedicated search and watch history tracking tables.

users (32 columns)

- **id** bigint unsigned, PRI
- **email** varchar(191), NOT NULL, UNI
- **username** varchar(191), NULL
- **password** varchar(191), NOT NULL
- **is_banned** tinyint, default: 0
- **is_subscribe** tinyint, default: 0
- **is_parental_lock_enable** tinyint(1), default: 0
- **pin** smallint, NULL
- **otp** smallint, NULL
- **user_type** varchar(191), NULL
- **producer_status** varchar(191), NULL
- **deleted_at** timestamp, MUL
- + 20 additional profile and banking columns

Key Supporting Tables

user_multi_profiles

11 cols — per-user profiles with child profile flag and avatar

devices

11 cols — device tracking with platform, session_id, active_profile

sessions

6 cols — web session store with IP, user_agent, payload

personal_access_tokens

10 cols — Sanctum tokens with abilities, expiry

tv_login_sessions

7 cols — TV session lifecycle management

web_qr_sessions

8 cols — QR auth with pending/authenticated/expired enum

2.2 Content Domain (18 Tables)

The content domain is the largest in the schema, with 18 tables governing the complete lifecycle of all streamable content — movies, TV series, standalone videos, episodes, seasons, clips, cast/crew, genres, and subtitles. The entertainments table at 59 columns serves as the authoritative content catalog entity, with rich support for TMDb integration, adaptive quality streaming, Bunny CDN delivery, SEO metadata, access control, and monetization fields co-located on the entity.



entertainments (59 cols)
Primary content catalog. Supports movies, TV shows, 360/180 video, TMDb sync, Bunny CDN, trailer URLs, approval workflow, per-content pricing and SEO metadata.



videos (54 cols)
Standalone video entities with independent quality tiers, download configuration, subtitle support, per-view revenue tracking, and Bunny CDN URL storage.



episodes (58 cols)
Episode-level entities linked to seasons and entertainments with independent streaming, download, subtitle, access control, and TMDb integration.



seasons (33 cols)
Series season entities with trailer, access, plan assignment, independent pricing, and SEO metadata including slug-based URL support.

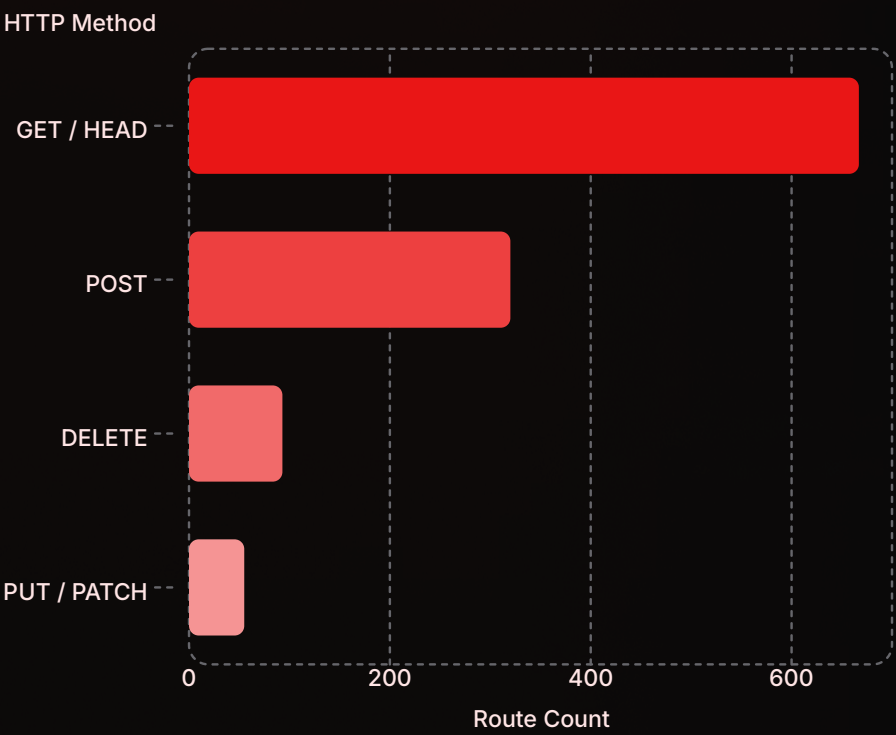
Stream & Download Mapping Tables

Mapping Table	Entity	Key Columns
entertainment_stream_content_mapping	Movies/Shows	entertainment_id, quality, url, type
video_stream_content_mapping	Standalone Videos	video_id, quality, url, type
episode_stream_content_mapping	TV Episodes	episode_id, quality, url, type
live_tv_stream_content_mapping	Live TV	tv_channel_id, stream_type, server_url

3. API Architecture

MXFlix exposes 1,135 routes across 4 access tiers. The REST API (/api prefix) provides 179 endpoints for mobile and third-party clients, authenticated via Laravel Sanctum. The platform implements URI-based versioning with unversioned, v1, v2, and v3 API tracks, enabling backward-compatible evolution of the mobile API surface while preserving legacy client compatibility.

HTTP Method Distribution



API Versioning Strategy

1	2
Unversioned Core auth, subscriptions, notifications, settings. Maximum backward compatibility.	v1 APIs Content acquisition workflows, genre listing, cast/crew, SEO endpoints.
3	4
v2 APIs Enhanced dashboard detail, top-10, tvshow list, web continue-watch.	v3 APIs Enhanced cast details, payment methods, app config, content-details, banner-data (25 endpoints).

3.1 Key API Endpoint Groups

The REST API surface covers all critical client operations including authentication, content discovery, playback, subscription management, social engagement, advertising, and device control. Below are the primary endpoint groups with their HTTP methods and named routes.

Authentication & Identity • POST /api/login • POST /api/register • POST /api/social-login • GET /api/send-otp • POST /api/verify-otp • POST /api/token-revoke • GET /api/logout-all	Content Discovery • GET /api/v3/content-list • GET /api/v3/content-details • GET /api/movie-details • GET /api/tvshow-list • GET /api/genre-content-list • GET /api/search-list • GET /api/coming-soon
Subscriptions & PPV • GET /api/plan-list • POST /api/save-subscription-details • POST /api/cancel-subscription • GET /api/ppv/movies • POST /api/save-payment-pay-per-view • GET /api/check-movie-purchase • GET /api/transaction-history	Engagement & Social • POST /api/save-likes • POST /api/save-rating • POST /api/save-watchlist • POST /api/save-continue-watch • POST /api/save-reminder • GET /api/watch-list • GET /api/notification-list

Authentication Tiers

1	2	3
Public APIs No authentication required — banners, app configuration, genre listing, onboarding data	Sanctum Bearer 87 endpoints protected by Sanctum token — account, subscriptions, content, social	Rate-Limited Sanctum + ThrottleRequests middleware — 48 protected endpoints with configurable limits

4. Authentication Deep Dive

MXFlix implements six distinct authentication methods covering web, mobile, API, and third-party access. This multi-modal approach ensures that every client surface — from admin dashboards to Flutter mobile apps to smart TVs — has an appropriate, secure authentication pathway. Each method is implemented through dedicated database tables and middleware guards.

EMAIL & PASSWORD

Admin & User login, API (Sanctum), **BCRYPT**.



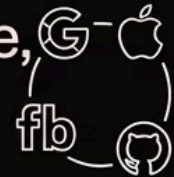
OTP-BASED AUTH

Send & verify OTP. Mobile passwordless.



SOCIAL LOGOIN (OAUTH)

Google, Apple, Facebook, GitHub.



SANCTUM API TOKEN

87 routes, Bearer token, personal tokens.



PRODUCER AUTH

Dedicated portal, CheckProducer middleware.



QR & TV AUTH

Mobile confirmation for web & TV sessions.



4.1 Authentication Methods Detail

1

Email & Password

Admin at `/admin/login`, **user** at `/login`, **API** at `POST /api/login`. **Bcrypt/Argon2** hashing. **Email verification** via `email_verified_at`. **Account suspension** via `is_banned` flag.

2

OTP Authentication

Passwordless via `GET /api/send-otp` and `POST /api/verify-otp`. **OTP** stored as `smallint` in `users.otp`. **Time-bound validity**, **mobile-first** flow with **web fallback** via `GET /otp-login`.

3

OAuth Social Login

Laravel Socialite integration. **Providers:** **Google, Apple, Facebook, GitHub**. `GET /login/{provider}` initiates, **callback** completes. **Provider data** persisted in `user_providers` (7 cols) with `provider` and `provider_id`.

4

Sanctum API Tokens

87 API routes protected. **Bearer token** issued on login, stored in `personal_access_tokens`. **Token abilities** for **granular scoping**. **Lifecycle** managed via `last_used_at` and `expires_at`.

5

Producer Portal

Dedicated registration and login at `/producer/login`. **CheckProducer** **middleware** **guards** **123** **exclusive routes**. **Status** tracked via `users.producer_status` column.

6

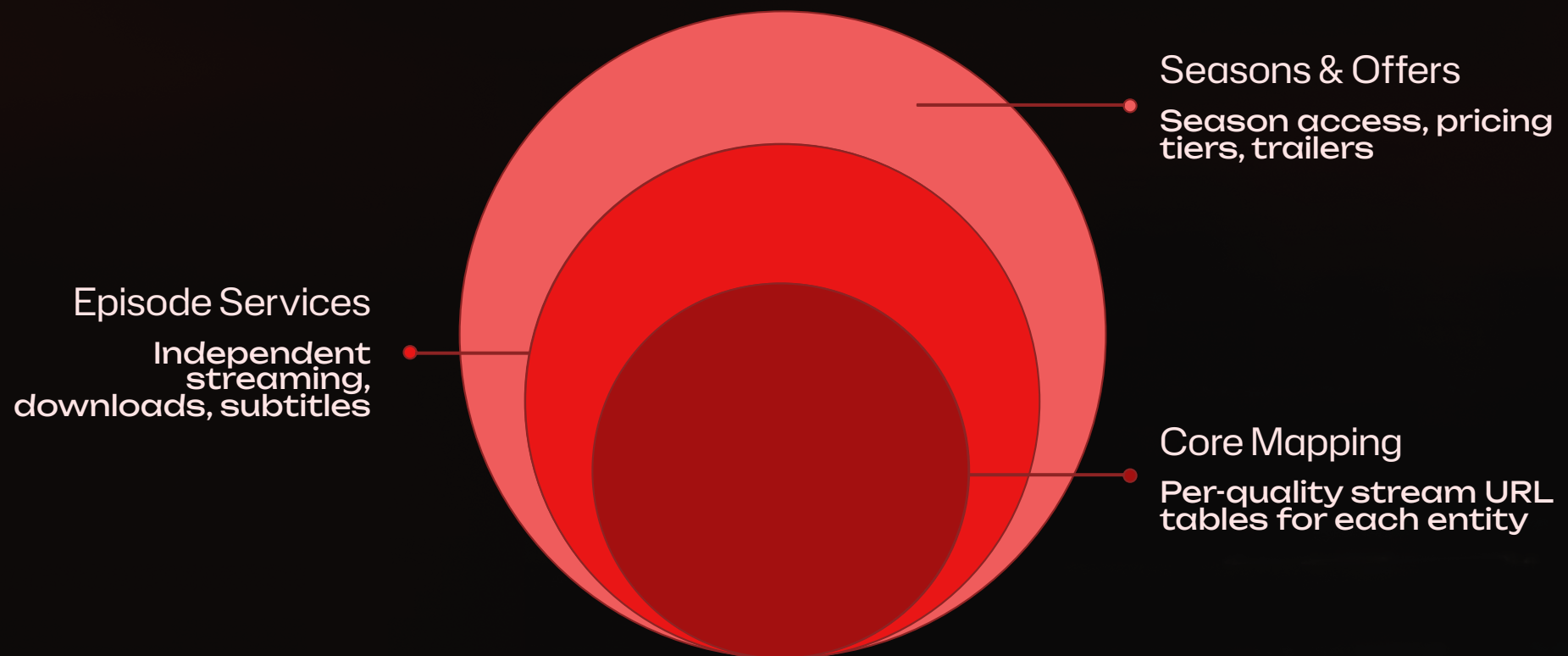
QR & TV Auth

`web_qr_sessions` (8 cols, enum: **pending/authenticated/expired**) and `tv_login_sessions` (7 cols) manage **second-screen** flows. **Mobile confirmation** via `POST /api/tv/confirm-session`.

5. Video Delivery Architecture

MXFlix supports multiple video delivery mechanisms across four content entity types: movies/shows (entertainments), standalone videos, episodic content, and live TV channels. The delivery architecture is built around quality-tier stream mapping tables that decouple content metadata from delivery URLs, enabling flexible CDN switching and multi-quality adaptive playback without schema changes.

Content Entity Hierarchy



Each level of the hierarchy carries its own independent streaming, download, subtitle, and access control configuration. This design supports complex access scenarios — for example, a series may be subscription-only while individual episodes can be purchased via PPV, and specific seasons may have different plan requirements.

Video Upload & Delivery Options



Direct Upload

Files uploaded directly to the platform, stored via Spatie Media Library, served from local disk or S3 with Bunny CDN as the delivery layer.



URL Input

External video URLs provided via `video_url_input`. Supports HLS, MP4, and embedded player streams for third-party hosted content.



Bunny CDN

Dedicated `bunny_video_url` and `bunny_trailer_url` columns on entertainments and episodes for Bunny Stream direct delivery with edge caching.



Live TV Streaming

`live_tv_stream_content_mapping` supports embedded players, dual server URLs, and configurable stream types per channel.

5.1 Subtitle & Download Architecture

Subtitle System

The platform implements a comprehensive multi-language subtitle system with the **dedicated subtitles table** (10 columns) supporting per-content subtitle tracks with language codes and default track selection.

- **subtitles:** `entertainment_id`, `language`, `language_code`, `subtitle_file`, `is_default`
- Content-level toggle: `enable_subtitle` flag on `entertainments`, `videos`, `episodes`
- Default subtitle: `is_default_subtitle` per content entity
- Inline subtitle file: `subtitle_file` for single-language content

Live TV Details

- 18-column `live_tv_channel` with `access` and `plan_id` control
- 12-column `live_tv_category` for channel organization
- 13-column stream mapping with embedded player support and dual server URLs
- Approval workflow on channels matching content entities

Offline Download System

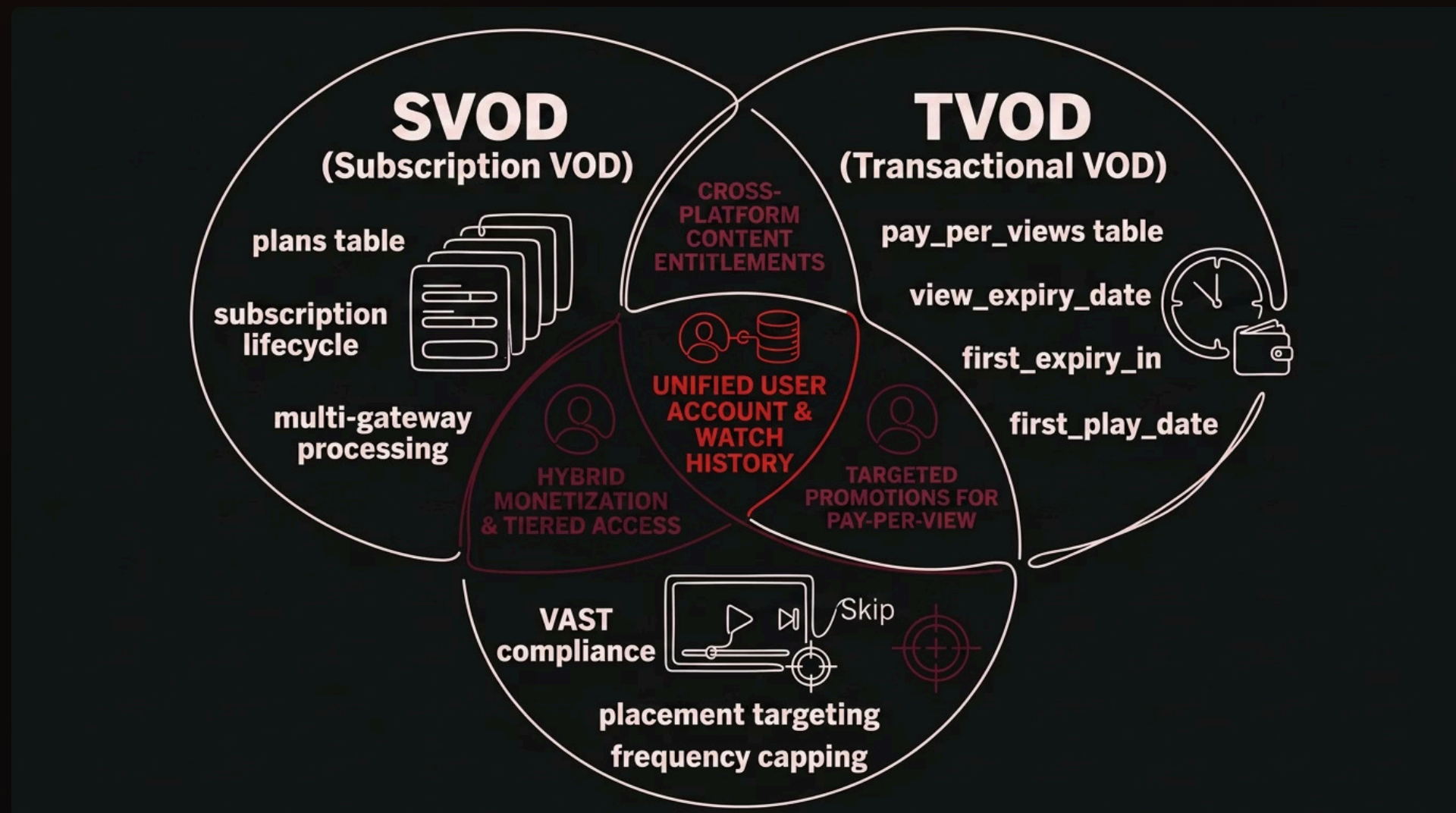
Four dedicated download mapping tables provide per-quality offline content delivery, each mirroring the stream mapping architecture with independent download URL management.

Download Table	Entity
<code>entertainment_download_mapping</code>	Movies/Shows
<code>video_download_mappings</code>	Standalone Videos
<code>episode_download_mapping</code>	TV Episodes

Download controls: `download_status` toggle, `enable_download_quality`, `download_type`, `download_url`. **API:** POST `/api/delete-download`, POST `/api/save-download`.

6. Monetization Technical Details

MXFlix supports three revenue models operating concurrently — SVOD, TVOD, and AVOD — enabling operators to maximize yield across different user segments and content types. The monetization engine is backed by 12 billing domain tables, a 7-gateway payment integration layer, a coupon system, multi-currency support, and a configurable tax engine. Revenue flows can be tracked from initial plan selection through payment processing to producer payout disbursement.



6.1 SVOD – Subscription Engine

The subscription system uses a three-table architecture: `plan` defines available tiers, `planlimitation_mapping` governs per-plan feature enforcement, and `subscriptions` tracks the full lifecycle of each user's active subscription. Platform and mobile store identifiers are stored separately to support Google Play, Apple App Store, and direct web payment flows from the same plan entity.

plan (20 columns)

- **identifier** / `android_identifier` / `apple_identifier` — platform store IDs
- **price**, `discount_percentage`, `total_price` — tiered pricing
- **level** — plan tier hierarchy for access control
- **duration** / `duration_value` — billing period configuration

Plan Limitation Features

- Maximum concurrent streams per plan
- Maximum devices per account
- Content access level restrictions
- Download quality limits

subscriptions (26 columns)

- **start_date** / `end_date` / `status` — lifecycle tracking
- **is_manual** — admin-assigned subscription flag
- **amount** / `discount_percentage` / `tax_amount` / `coupon_discount` / `total_amount`
- **device_id** — per-device subscription binding
- **payment_id** — transaction linkage
- **plan_type** — JSON plan snapshot at time of purchase
- **level** — denormalized access tier



Manual subscription creation via `is_manual` flag enables admin-assigned plans for promotional or enterprise customers without payment gateway flow.

6.2 TVOD, AVOD & Payment Gateways

Pay-Per-View (TVOD)

The `pay_per_views` table (13 columns) enables content-level purchases with time-limited access windows. Key fields include `first_play_date`, `view_expiry_date`, `access_duration`, and `available_for` for flexible rental window configuration. API: `GET /api/check-movie-purchase`, `GET /api/check-episode-purchase`.

Advertising (AVOD)

Two independent ad systems: VAST-compliant ad server integration (`vast_ads_setting`, 17 cols) and custom self-served ads (`custom_ads_setting`, 20 cols). Both support skip controls, frequency capping, date-bounded campaigns, and content/category targeting.

Payment Gateway Matrix

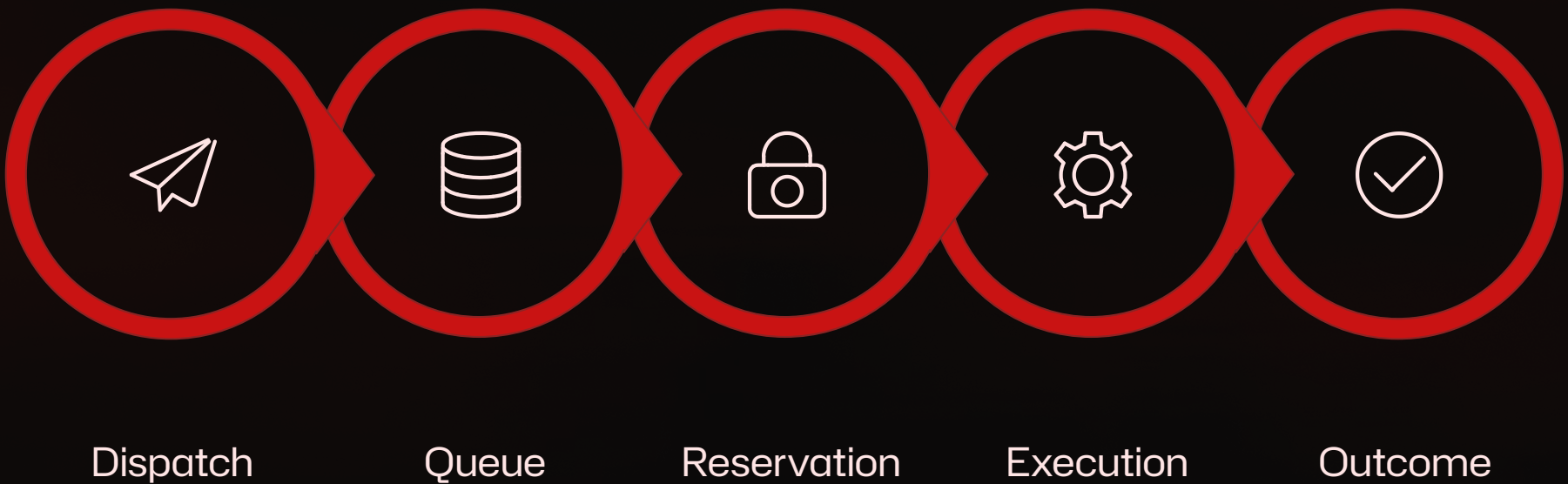
Stripe Global — Cards, Apple Pay, Google Pay	Razorpay India — UPI, Cards, Net Banking
PayPal Global — PayPal accounts, Cards	Paystack Africa — Cards, Bank transfers
Flutterwave Africa — Cards, Mobile money	Midtrans SE Asia — E-wallets, Bank transfer
PhonePe India — UPI-based payments	

Coupon System

The coupon system provides percentage and fixed discount types via the `coupons` table (14 cols, unique code constraint). Plan-specific coupon targeting is managed through `coupon_subscription_plan`. Redemption is tracked in `user_coupon_redeem` with the applied discount amount captured for financial reconciliation. Coupons are date-bounded via `start_date/expire_date`.

7. Background Processing & Queues

MXFlix leverages the Laravel queue system with Redis as the backing driver for asynchronous processing across six job categories: video processing, email notifications, push notifications, content import, analytics aggregation, and subscription lifecycle management. Laravel Horizon provides real-time queue monitoring and worker management via a web dashboard.



Job Categories

- Video processing: transcoding, thumbnails, quality conversion
- Email: welcome, subscription, password reset
- Push: Firebase FCM dispatch for mobile
- Import: TMDb sync, bulk content operations
- Analytics: watch log processing, revenue calculations
- Subscriptions: renewal, expiry, downgrade handling

Worker Configuration

- 3–5 workers standard; 10+ for high-traffic
- Retry limit: 3 attempts before failed_jobs
- Timeout: 60s standard, 300s video processing
- Sleep: 3 seconds between polling cycles
- Max jobs: 1,000 per worker before restart

Batch Processing

The `job_batches` table (10 cols) tracks aggregate progress across grouped operations with `total_jobs`, `pending_jobs`, `failed_jobs`, and `finished_at` for coordinated parallel workflows.

Scheduled Tasks

- Subscription expiry checking and auto-renewal
- Session cleanup (web, TV, QR sessions)
- Notification campaign dispatch
- Analytics aggregation and reporting
- Failed job cleanup and archival
- Cache warming for configuration data

8. Storage Architecture

MXFlix supports multiple storage backends for media and assets, governed by the Spatie Media Library integration via the `media` table (19 cols). The polymorphic association model allows any entity to have associated media without schema coupling. Producer-scoped media isolation is enforced via `producer_id` on both the `media` and `filemanagers` tables, ensuring content partners cannot access each other's assets.



AWS S3

Primary production media storage backend. High durability (99.999999999%), integrated with CDN for origin fetch. Configured as the default Spatie Media Library disk for uploaded content.



Bunny CDN

Video delivery with edge caching optimized for streaming. Dedicated `bunny_video_url` columns on `entertainments` and `episodes` allow direct Bunny Stream URLs bypassing the `media` table entirely.



Local Storage

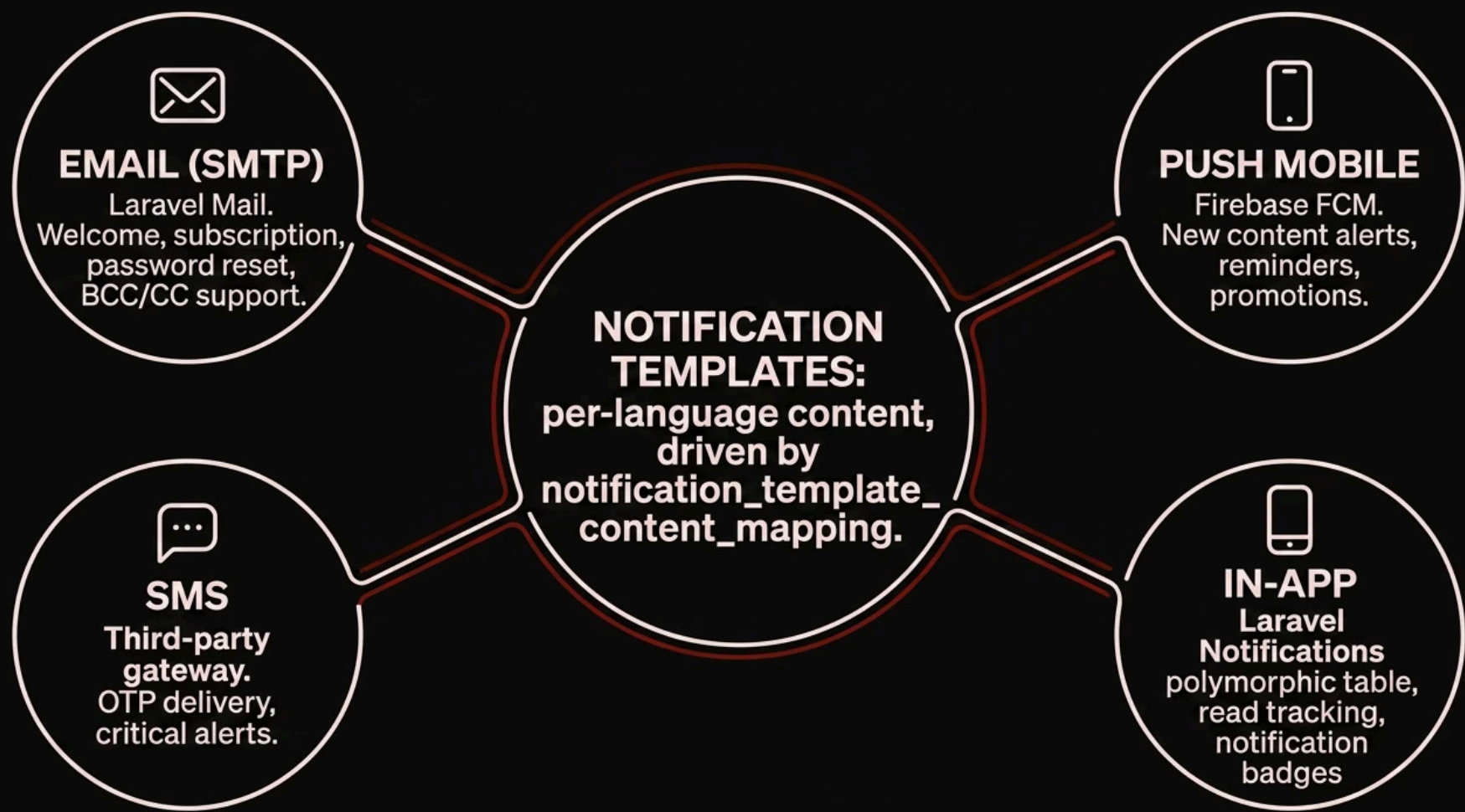
Development and single-server deployment backend. Direct file system access via Laravel's local disk driver. Not recommended for production due to scalability constraints.

File Manager Architecture

The producer-facing file manager provides a hierarchical folder structure via the `folders` table (11 cols) with self-referencing `parent_id` for unlimited depth tree structures. The `filemanagers` table (10 cols) stores individual file records with producer isolation and full soft-delete support. This enables content partners to organize their media assets independently with consistent access control enforcement.

9. Notification System Architecture

MXFlix implements a multi-channel notification system powered by Laravel Notifications with a template engine that supports per-language content mapping, user-type targeting, and multi-channel delivery. The notification architecture covers four delivery channels — email, mobile push, SMS, and in-app — all driven by configurable templates stored in the database.



Template System Details

`notification_templates` (16 cols)

- Template type classification and channel specification
- BCC and CC support for broadcast notifications
- `to` field for direct recipient targeting
- Status toggle for template activation

`notification_template_content_mapping` (17 cols)

- Per-language content variants for i18n
- User type targeting (admin, producer, subscriber)
- Email subject and notification subject separation
- Deep link support via `notification_link` field

10. Analytics & Reporting Architecture

MXFlix provides comprehensive analytics through dedicated tracking tables covering watch behavior, revenue attribution, content engagement, and a full audit log. The analytics architecture is built for post-processing via queue workers rather than real-time aggregation, enabling high-throughput write paths for watch events without impacting query performance on reporting surfaces.

Watch Analytics <code>video_watch_logs</code> (15 cols): <code>seconds_watched</code>, <code>watch_date</code>, profile-level tracking, payout linkage. <code>continue_watch</code>: resume position with <code>watched_time</code> and <code>total_watched_time</code>.	Revenue Analytics Subscription revenue via <code>subscriptions_transactions</code> with <code>tax_data</code>. PPV revenue via <code>payperviewstransactions</code>. Per-view revenue on videos (decimal 10,4). Producer payouts table.
Content Analytics Acquisition lifecycle via <code>content_acquisition_logs</code>. Search behavior via <code>user_search_histories</code>. Engagement via likes, reviews, watchlist. <code>entertainment_views</code> for view counts.	Audit Logging Spatie ActivityLog: polymorphic subject/causer tracking, JSON change snapshots, batch UUID grouping, event classification. Covers all 57 audited tables.

activity_log (12 columns)

Column	Type	Purpose
<code>log_name</code>	<code>varchar(191)</code> , MUL	Log channel classification
<code>subject_type</code> / <code>subject_id</code>	<code>varchar(191)</code> , bigint, MUL	Polymorphic entity being changed
<code>causer_type</code> / <code>causer_id</code>	<code>varchar(191)</code> , bigint, MUL	Actor performing the change
<code>event</code>	<code>varchar(191)</code>	Change classification (created/updated/deleted)
<code>properties</code>	<code>longtext</code>	JSON before/after change snapshot
<code>batch_uuid</code>	<code>char(36)</code>	Groups related changes into a batch

11. Mobile Application Architecture

MXFlix provides Flutter-based mobile applications delivering a single codebase across iOS, Android, Web, and Desktop platforms. The mobile architecture is tightly coupled to the REST API surface, consuming 179 versioned endpoints via Sanctum token authentication. Mobile-specific configuration is managed through the `mobile_settings` table, enabling runtime behavior changes without app redeployment.



Flutter Cross-Platform

Single Dart codebase for iOS, Android, Web, and Desktop. Consumes 179 REST API endpoints via Sanctum Bearer token auth with JSON response format following Laravel API conventions.



Parental Controls

`users.pin (smallint)` for PIN lock.
`is_parental_lock_enable` toggle. API: POST `/api/update-parental-lock` and POST `/api/verify-pin` for PIN enforcement on restricted content.



Multi-Profile Support

`user_multi_profiles`: per-user profiles with name, avatar, and `is_child_profile` flag for kids-safe content filtering. Per-profile watch history, continue watching, and watchlists maintained independently.

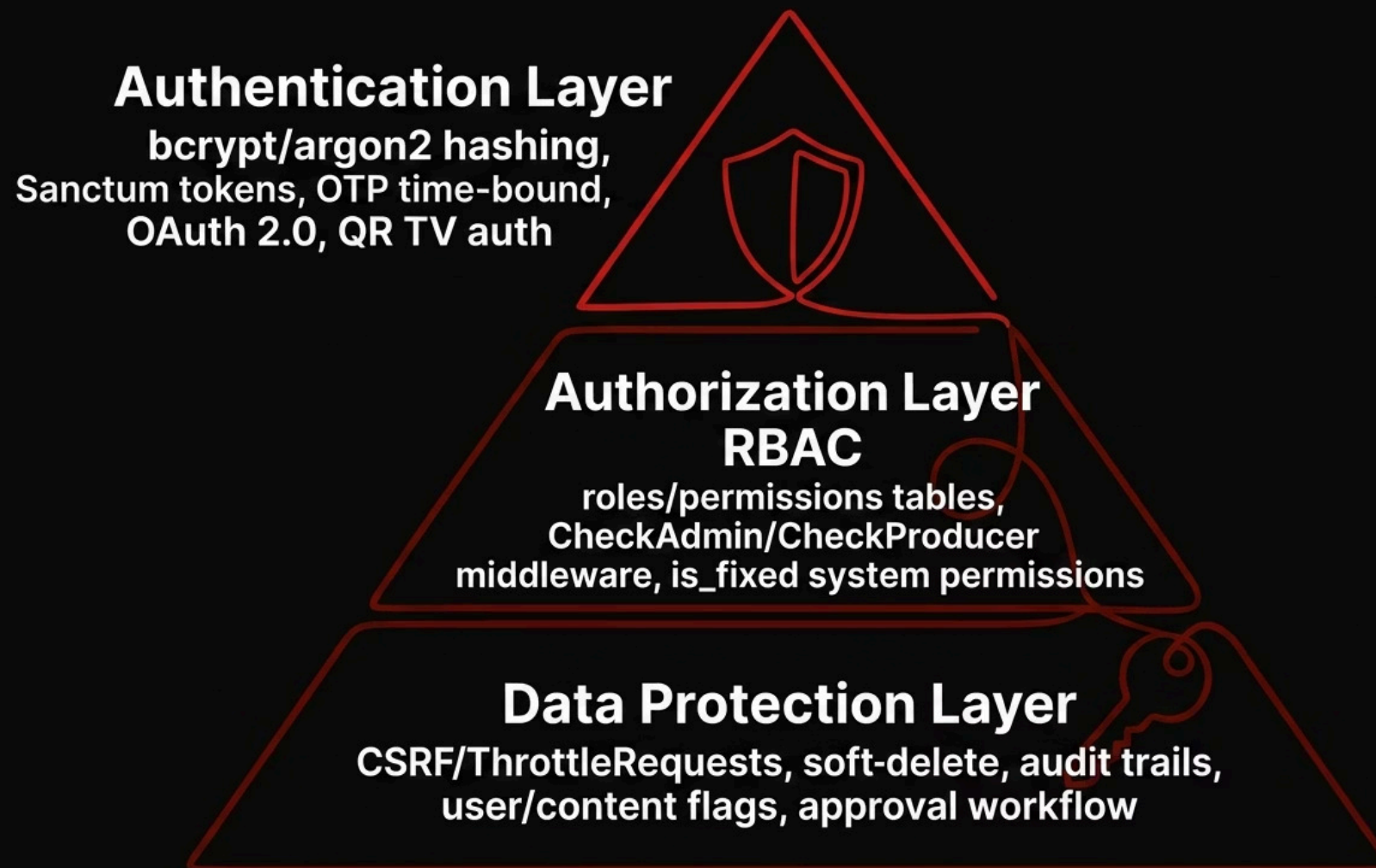


Offline Downloads

Quality-selectable offline content download. Separate download mapping tables per content type. POST `/api/save-download` tracks downloads; POST `/api/delete-download` manages cleanup.

12. Security Architecture Summary

MXFlix implements multi-layer security controls across authentication, authorization, and data protection. The security posture is built on proven Laravel ecosystem packages — Sanctum for API auth, Spatie Permission for RBAC, and the standard Laravel middleware pipeline for CSRF, rate limiting, and route-level access control.



12.1 RBAC Authorization Details

Role-Based Access Control is powered by Spatie Laravel Permission with a five-table schema governing role definitions, permission definitions, role-to-permission mapping, and polymorphic model-level assignments. The `is_fixed` flag on both roles and permissions prevents accidental deletion of system-critical access control entries. The `guard_name` field supports multiple authentication guards (web, API) with independent permission scopes.

RBAC Schema

- **roles** (7 cols): name, title, guard_name, is_fixed
- **permissions** (6 cols): name, guard_name, is_fixed
- **role_has_permissions** (2 cols): composite PK (permission_id, role_id)
- **model_has_roles** (3 cols): polymorphic role assignment
- **model_has_permissions** (3 cols): direct model permission grants

Access Control Enforcement

- **CheckAdmin middleware** guards all 647 admin routes with role validation
- **CheckProducer middleware** guards 123 producer portal routes exclusively
- **Sanctum middleware** protects 87 API routes with bearer token validation
- **ThrottleRequests** rate-limits 48 API endpoints against abuse
- **Content approval workflow** (pending/approved/rejected) prevents unauthorized publication

13. White-Label & Extensibility

MXFlix is architected for white-label deployment with extensive runtime customization options backed by database-driven configuration. The settings, constants, and languages tables form the core configuration engine, allowing platform operators to modify behavior, branding, and UI strings without code changes or redeployment. The modular architecture adds feature-level toggleability through the modules table.

Configuration System

- **settings**: dynamic key-value with type and datatype classification
- **constants**: categorized constants with type/sub_type and sequence ordering
- **languages**: UI string localization (English, Arabic, Greek, French, German)
- **mobile_settings**: runtime mobile app behavior without redeployment

Branding Customization

- **onboardings**: onboarding slide content and imagery
- **banners** (21 cols): image/video banners with media_type enum
- **seo**: platform-level meta_title, meta_keywords, canonical_url
- **pages**: slug-based CMS pages for legal and informational content
- **faqs**: self-service FAQ management

Module Toggleability

- **modules** (9 cols): module_name, status (on/off toggle)
- **is_custom_permission**: per-module permission control
- **more_permission**: extended capability assignment per module

Geographic Configuration

- countries, states, cities, worlds for regional deployment
- **entertainment_country_mapping**: geo-restricted content availability
- **currencies** (15 cols): multi-currency with symbol, position, decimal format
- **taxes**: percentage and fixed tax type per jurisdiction

14. Performance Characteristics

MXFlix is designed for high-performance streaming delivery with multiple optimization layers spanning caching strategy, database design, queue throughput, and frontend asset delivery. The Redis-first approach to caching, sessions, and queues minimizes database pressure on the hot path, while the CDN integration strategy offloads video bandwidth entirely from the application tier.



Database Optimization

Indexed columns on deleted_at, subject_type, causer_type. Normalized 94-table schema minimizing duplication. Soft-delete avoids expensive DELETE operations. bigint unsigned PKs for large-scale data growth.



Caching Strategy

Redis cache driver for sessions, application data. Route caching and config caching reduce bootstrap time. CDN integration (Bunny, AWS CloudFront) for media delivery at edge with sub-100ms TTFB targets.



Queue Performance

Redis-backed queue for high-throughput job processing. Laravel Horizon for real-time monitoring. Job batching for parallel processing of bulk operations. Failed job retry with exponential backoff.



Video Delivery

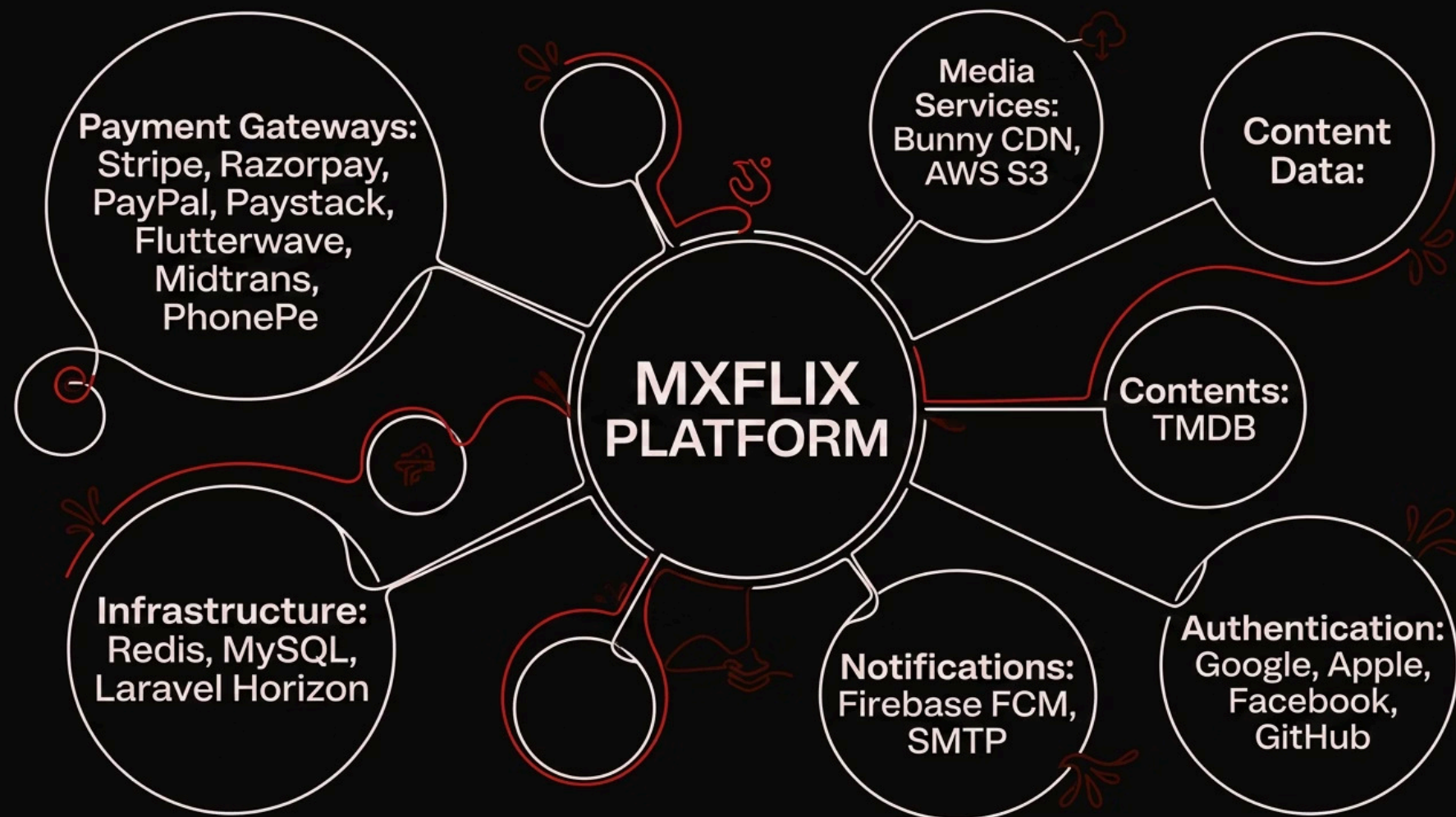
Multi-quality streaming (360p to 4K) via stream content mapping. Bunny CDN edge delivery. Adaptive bitrate streaming support. Separate poster_tv_url for TV versus web display optimization.

Scalability Roadmap



15. Integration Reference

MXFlix integrates with 20+ external services across payments, media delivery, push notifications, content metadata, and social identity. All payment gateways are registered in the `service_providers` table (15 cols) providing centralized provider management with manager assignment and contact information. Webhook infrastructure enables event-driven integration with external systems.



Webhook System

The `webhook_calls` table (8 cols) provides outgoing webhook infrastructure with named webhooks, custom URLs, configurable headers, payload storage for audit, and exception logging for delivery failure analysis. This supports event-driven integrations with external billing systems, analytics platforms, and content management workflows.

16. Operational Requirements

MXFlix deployment requires a well-configured LAMP/LEMP stack with Redis as a mandatory dependency for queues, caching, and session management. The installer middleware (CheckInstallation) enforces prerequisite validation on first run, while the Supervisor-managed queue workers and Laravel scheduler cron job handle ongoing background processing requirements.

Component	Minimum Specification
Web Server	Apache 2.4+ or Nginx (latest stable)
PHP	8.2+ with required extensions (BCMath, Ctype, cURL, DOM, Fileinfo, JSON, Mbstring, OpenSSL, PCRE, PDO, Tokenizer, XML)
MySQL	8.0+ with binary logging enabled for PITR
Redis	6.0+ for caching, sessions, and queue driver
Storage	AWS S3 or Bunny CDN account configured
SSL	TLS certificate for HTTPS (HSTS recommended)
Cron	Laravel scheduler access (crontab every minute)

Monitoring & Backup

Monitoring Stack

- Laravel Horizon for queue monitoring and worker metrics
- Laravel Telescope for development debugging
- Sentry-compatible error tracking integration
- Slow query logging for database performance
- Uptime monitoring with health check endpoint

Backup & Recovery

- Automated MySQL backups (daily recommended)
- Redis persistence: RDB snapshots + AOF
- Media backup to secondary S3 bucket or Bunny CDN
- Configuration files under version control
- Binary log (binlog) position for point-in-time recovery

17. Technical Risk Assessment

The following risk assessment covers scalability, maintenance, data integrity, extensibility, regulatory compliance, and technical debt dimensions. Each risk is characterized by likelihood and impact to support infrastructure planning and mitigation prioritization. The assessments are derived from direct schema and architecture analysis without speculative inference.

Risk Matrix

Risk Category	Likelihood	Impact	Mitigation
Video delivery bandwidth spike	Medium	High	CDN auto-scaling, adaptive bitrate
Database performance at scale	Low	High	Read replicas, query optimization
Payment gateway downtime	Low	High	Multi-gateway fallback routing
Content piracy / unauthorized sharing	Medium	Medium	DRM integration, watermarking
Queue worker failure cascade	Low	Medium	Circuit breaker, job isolation
Data center outage	Low	Critical	Multi-region deployment, DR site

17.1 Regulatory & Technical Debt Considerations

Regulatory Considerations

- **GDPR:** User data management and deletion capabilities via soft-delete and account deletion API (POST /api/delete-account)
- **Content Licensing:** Managed via content acquisition workflow (draft → submitted → approved → contracted → published → expired)
- **Revenue Reporting:** Full transaction records in subscriptions_transactions and payperviewtransactions for tax compliance
- **Child Safety:** Parental controls via PIN lock + is_child_profile flag + is_restricted content flag + content_rating field
- **Audit Trail:** 57-table activity logging for regulatory inquiry response via Spatie ActivityLog

Technical Debt Assessment

Architecture Extraction

Monolithic architecture may require service extraction for extreme scale (10M+ subscribers). Module boundaries provide extraction readiness.

Session Storage

Sessions stored in MySQL sessions table; Redis-only sessions recommended at scale to eliminate database pressure on the hot path.

FK Constraints

Some tables lack explicit foreign key constraints at the database level; integrity enforced at the ORM level only. Consider adding DB-level constraints for critical relationships.

Test Coverage

Queue worker and payment integration test coverage could be expanded. Recommend unit tests for subscription lifecycle and PPV access validation.

❏ **The content acquisition state machine** (draft → submitted → under_review → approved → rejected → contracted → published → expired) represents a strong foundation for IP rights management and should be evaluated as the basis for DRM workflow integration in regulated markets.

Document Summary & Conclusions

MXFlix represents a feature-complete, enterprise-grade OTT streaming platform with a well-structured modular monolith architecture, comprehensive API surface, multi-modal authentication, concurrent SVOD/TVOD/AVOD monetization, and production-ready operational infrastructure. The 94-table schema demonstrates thorough domain modeling with consistent soft-delete, audit, and SEO patterns applied uniformly across all content and user entities.

Architecture

Laravel 12, PHP 8.2+, modular monolith with 20+ independent domain modules, Redis-backed queues, Flutter mobile clients

Data Model

94 tables, 1,229 columns, 67 soft-delete tables, 57 audited tables, 9 functional domains with consistent design patterns

API Surface

1,135 routes, 179 REST API endpoints, URI versioning (v1/v2/v3), Sanctum authentication, 48 rate-limited endpoints

Monetization

Concurrent SVOD, TVOD, AVOD — 7 payment gateways, coupon system, multi-currency, configurable taxation

MXFlix Technical Product Dossier — Generated from live schema and route analysis. Document version: 1.0 | Classification: Technical Evaluation