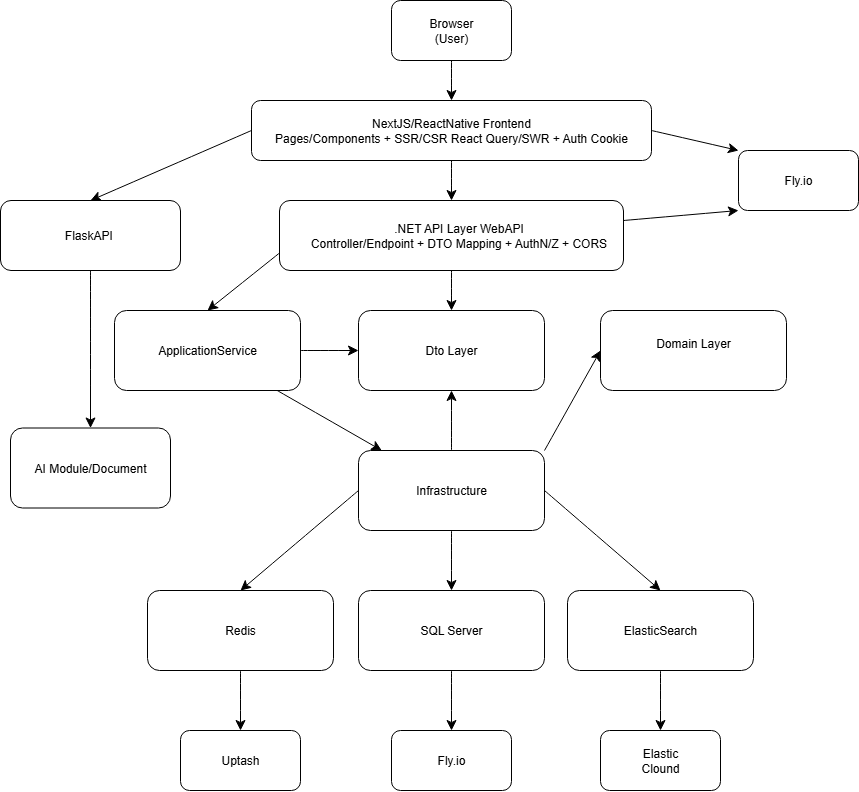
FilmZone — Next.js/React Native + .NET Clean Architecture (Integrated)

Bản mô tả hợp nhất kiến trúc Web/Mobile (.NET Clean Architecture) và dịch vụ AI (FlaskAPI) trong một tài liệu.

# Sơ đồ tổng thể (tích hợp FlaskAPI)



# 1) Tổng quan

Hệ thống gồm hai frontend: Next.js (Web, SSR/CSR) và React Native (Mobile), gọi .NET Backend tuân thủ Clean Architecture (API → ApplicationService → Domain; Infrastructure làm adapter). Redis (Upstash) dùng cho cache và hàng đợi; Elasticsearch (Elastic Cloud) cho tìm kiếm/full‑text và gợi ý; SQL Server chạy trên Fly.io với volume. Dịch vụ AI viết bằng Python (FlaskAPI) xử lý trích xuất phụ đề từ video và gợi ý phim. CI/CD: GitHub Actions → Docker → Fly.io.

# 2) Thành phần & Vai trò

• Frontend (Next.js/React Native): SSR/CSR, React Query/SWR; Web dùng cookie HTTP‑only; Mobile dùng Bearer token + Secure Storage.

• .NET API Layer: Controllers/Endpoints, DTO Mapping, AuthN/Z, CORS, ProblemDetails; version hoá API (/v1).

• ApplicationService (Use Cases): điều phối nghiệp vụ, validation, transaction boundary, phát Outbox event.

• Domain Layer: Entities, Value Objects, Domain Events/Rules; aggregate Movie–Episode bảo toàn bất biến.

• Infrastructure: EF Core/DbContext & Repositories, Redis Cache Adapter, Elasticsearch Adapter, File Storage Adapter, Outbox Dispatcher.

• DB/Services: SQL Server (Fly.io, private network), Upstash Redis (TLS rediss://), Elastic Cloud (API Key), S3/Blob (media).

• FlaskAPI (Python): microservice AI cho phụ đề & gợi ý

# 3) Luồng nghiệp vụ điển hình

## 3.1 Thêm Episode

• UI (Next.js/React Native) gửi POST /api/movies/{id}/episodes (JSON; Web dùng cookie HTTP‑only).

• API map DTO → CreateEpisodeCommand, gọi Use Case.

• Use Case kiểm tra quy tắc domain (thứ tự season/episode, duration, trạng thái publish).

• Use Case gọi IEpisodeRepository/IMovieRepository (EF Core) → ghi DB trong 1 transaction.

• Phát Outbox event để index search/cache invalidation.

• Trả 201 Created → Frontend cập nhật cache/state và điều hướng.

## 3.2 Tạo phụ đề tự động (FlaskAPI)

• API .NET nhận yêu cầu tạo phụ đề → phát Outbox `SubtitleRequested` hoặc gọi thẳng FlaskAPI (tùy use case).

• FlaskAPI tải video (pre‑signed URL), FFmpeg trích audio → ASR (Whisper/faster‑whisper).

• (Tuỳ chọn) Diarization bằng pyannote.audio; chuẩn hoá dấu câu & timestamp → sinh SRT/WebVTT + segments.json.

• Upload output lên Storage; ghi metadata vào DB/Elastic; FlaskAPI bắn webhook `SubtitleReady` về .NET hoặc cho phép client polling.

• .NET cập nhật trạng thái và lưu URL phụ đề; Frontend hiển thị và cho phép bật subtitle.

## 3.3 Gợi ý phim (Hybrid)

• Batch embedding: FlaskAPI tạo embedding từ title/genres/plot (sentence‑transformers) → index `dense\_vector` vào Elasticsearch.

• Realtime: API .NET gọi FlaskAPI /ai/recommendations (hoặc truy vấn ES trực tiếp) để lấy top‑K đề xuất; kết hợp tín hiệu hành vi (watch/like).

• Ghi feedback: client gửi /ai/feedback để cải thiện gợi ý; cache kết quả ngắn hạn bằng Redis.

# 4) API Contracts (trích yếu, gộp FlaskAPI)

• .NET: POST /api/movies/{id}/episodes, GET /api/movies/{slug}, v.v…

• FlaskAPI:

POST /ai/subtitles/jobs — tạo job phụ đề

Request JSON:  
{  
 "video\_url": "https://.../movie.mp4",  
 "language": "auto",  
 "translate\_to": "vi",  
 "diarize": true,  
 "output": ["srt","vtt","json"],  
 "webhook\_url": "https://api.yourapp.com/ai/subtitles/callback"  
}

GET /ai/subtitles/jobs/<job\_id> — lấy trạng thái/kết quả

POST /ai/recommendations — gợi ý theo movie/user/context

Request JSON:  
{  
 "current\_movie\_id": "mv\_1024",  
 "user\_id": "u\_789",  
 "top\_k": 10,  
 "filters": {"genres": ["action"]}  
}

# 5) CI/CD (GitHub Actions → Docker → Fly.io)

• Build & Test .NET (dotnet restore/build/test); Dockerize API; deploy với flyctl (`filmzone-api`).

• Build FlaskAPI (Docker, gunicorn/flask); deploy app Python với flyctl (`filmzone-ai`).

• Secrets: FLY\_API\_TOKEN, CONNECTIONSTRINGS\_\_DEFAULT, REDIS\_URL, ELASTIC\_URI, ELASTIC\_API\_KEY, STORAGE\_\*, ASR\_MODEL…

• SQL Server deploy riêng (`filmzone-sql`) với volume; API kết nối qua host nội bộ `filmzone-sql.internal:1433`.

# 6) Bảo mật & Quan sát

• Web: cookie HTTP‑only + SameSite; Mobile: Secure Storage + Bearer token + refresh rotation; CORS đúng origin.

• Redis qua rediss:// (TLS). SQL Encrypt=True; Trust Server Certificate=True. Elastic dùng API Key, rotate định kỳ.

• Webhook từ FlaskAPI ký HMAC (X‑Signature) và kiểm tra timestamp chống replay.

• Health endpoints: /health & /ready; logging Serilog; tracing OpenTelemetry; metrics Prometheus.

# 7) Checklist triển khai

• Tạo 3 app Fly: filmzone-api (.NET), filmzone-sql (SQL + volume), filmzone-ai (FlaskAPI).

• Thiết lập Upstash Redis và Elastic Cloud; lưu URI/API Key vào secrets.

• Khai báo biến môi trường cho cả hai services (.NET & Flask).

• Thêm Outbox dispatcher và worker; test webhook end‑to‑end cho phụ đề.