Streaming Agent

오버레이 기반 스트리밍 데이터 수집 플랫폼
사업 계획서

플랫폼 의존에서 벗어나
게임 중심의 스트리밍 인사이트를 확보하는 방법

넥슨 크리에이터파트너십팀

김인주

내부 전략 제안서

2026년 1월

Executive Summary

데이터 독점 확보	ROI 측정 가능	생태계 지배력
전 플랫폼 실시간 수집	캠페인 성과 정량화	스트리밍 인프라 구축

Streaming Agent는 스트리머 방송 화면 위에서 동작하는 오버레이 기반 에이전트로, 멀티 플랫폼(SOOP, 치지직, YouTube, Twitch)의 방송·채팅·후원·이벤트 데이터를 실시간으로 수집·정규화·분석하는 데이터 인입 시스템이다.

현재 스트리밍은 게임 마케팅에서 매우 중요한 채널임에도 불구하고, 데이터는 플랫폼에 종속되어 있으며, 스트리머 운영과 캠페인 성과 분석은 감각과 경험 중심으로 이루어지고 있다. 플랫폼은 법률적·사업전략적 이유로 타 게임, 타 카테고리 데이터, 후원금 정보를 제공하지 않는다.

본 사업은 플랫폼과 경쟁하지 않는다. **모든 이벤트가 통과하는 '툴(오버레이)' 레이어를 확보함으로써 데이터 주도권을 획득**하고, 스트리머-유저-게임을 하나의 데이터 루프로 연결하는 것을 목표로 한다.

핵심 전략:

1. 모든 스트리머가 쓰는 오버레이를 만들어 **결집**시킨다
2. 그 구조 위에서 넥슨이 **데이터를 수집할 수 있는 파이프라인**을 확보한다

현재 시스템 현황: 35개 데이터베이스 테이블, 50+ REST API 엔드포인트, 10종 오버레이, 실시간 Socket.io 이벤트 파이프라인 구축 완료.

사업 모델: 모든 기능을 무료로 제공하여 스트리머 기반을 빠르게 확보하고, 오버레이 내 광고 인벤토리를 통한 수익화를 검토 중. 서비스 자체의 영리보다 데이터 확보를 통한 넥슨 전체 사업의 의사결정 품질 향상이 핵심 가치.

문서 구성: 본 문서는 **Part I: 사업 전략** (사업 배경, 데이터 전략, 수익 모델, 리스크)과 **Part II: 기술 명세** (시스템 아키텍처, 데이터 스키마, API)로 구성되어 있다.

차례

그림 차례

표 차례

제 I 편

사업 전략

제 1 장

사업 배경 및 문제 정의

1.1 스트리밍과 게임 마케팅의 관계

스트리밍 사업군은 게임에 직접적인 효과가 있는가? 이 질문에 대한 답은 명확히 "예"이다. 라이브 스트리밍은 게임 마케팅의 핵심 채널로 자리잡았으며, 시청자의 게임 구매 결정, 커뮤니티 형성, 브랜드 인지도에 직접적 영향을 미친다.

그러나 문제는 **이 효과를 측정할 수 없다**는 점이다. 현재 넥슨이 스트리밍 마케팅에 투자하는 비용 대비 실제 효과를 정량적으로 분석할 수 있는 데이터 인프라가 존재하지 않는다.

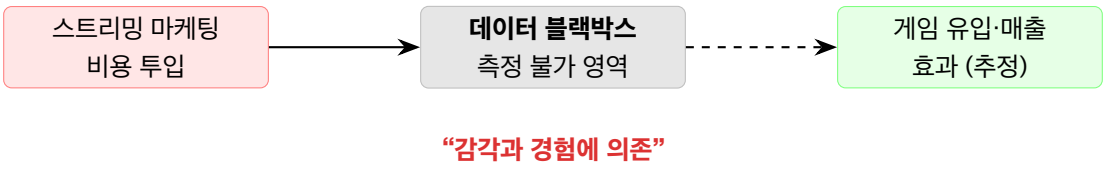

그림 1.1: 현재 넥슨의 스트리밍 마케팅 의사결정 흐름

1.2 현재 직면한 문제

스트리밍 사업을 분석하고, 넥슨 게임이 인터넷 방송 신(scene)에서 잘 굴러가게 하려면, 우리는 먼저 "어떤 데이터가 필요한가"를 정의해야 한다. 그리고 현재 우리가 그 데이터를 갖고 있지 못한 이유를 직시해야 한다.

표 1.1: 넥슨이 직면한 3가지 구조적 문제

문제	상세 설명
직관에 의존한 민심 파악	숙제 방송이 아닌 곳에서 어떤 게임이 인기를 얻고 있는지, 유저들이 우리 게임에 대해 어떻게 말하는지 파편화된 정보에 의존해 추측.
데이터 주권의 부재	모든 데이터 주권은 각 플랫폼에 종속. 그들이 제공하는 제한적인 데이터만 수동으로 확인 가능.
측정 불가한 ROI	데이터 부재는 곧 성과 측정 부재. 스트리밍 마케팅 비용이 실제로 얼마나 효과가 있었는지 답할 수 없음.

1.3 플랫폼이 데이터를 주지 않는 이유

스트리밍 데이터를 가져오기 위해 플랫폼에 물어보기 시작했으나, 플랫폼 입장에서는 다음과 같은 이유로 데이터 제공이 어렵다.

- **법률상 문제**: 개인정보보호법에 의해 시청자 행동 데이터의 제3자 제공에 제약.
- **사업전략상 문제**: 타 게임, 타 카테고리 데이터는 플랫폼의 핵심 자산. 경쟁사에 유출될 위험.
- **후원금 데이터**: 플랫폼의 수익 모델 핵심. 외부 공개 시 사업 모델 노출.

표 1.2: 데이터 수집 시도와 한계

시도	결과	한계
공식 API 연동	제한적 성공	실시간 불가, 카테고리 데이터 없음
제휴/협업 논의	진행 지연	플랫폼 이해관계 충돌
크롤링/간접 수집	법적 리스크	정책 위반 우려, 불안정

결론: **모든 방법은 결국 플랫폼 허락 없이는 불가능하다.** 이는 구조적 한계이며, 접근 방식 자체를 전환해야 한다.

제 2 장

분석에 필요한 데이터 정의

2.1 데이터 수요 분석

넥슨이 스트리밍 생태계에서 답해야 할 핵심 질문과, 그에 필요한 데이터를 매핑한다.

표 2.1: 사업 질문과 필요 데이터 매핑

사업 질문	필요 데이터	수집 테이블
어떤 스트리머가 우리 게임에 진정한 영향력이 있는가?	시청자 참여율, 후원 금액, 채팅 밀도	viewer_engagement, events, broadcasts
경쟁 게임 대비 우리 게임의 방송 비중은?	카테고리별 방송 수, 시청자 수	platform_categories, category_stats
캠페인 진행 시 실제 시청자 반응은?	실시간 채팅 감성, 후원 급증 패턴	events, chat_stats, viewer_snapshots
어떤 스트리머를 새로 발굴해야 하는가?	성장률, 시청자 충성도, 카테고리 다양성	persons, viewer_engagement, broadcast_segments
타 게임 스트리머를 넥슨으로 유입할 수 있는가?	타 카테고리 방송 데이터, 시청자 이동 패턴	unified_games, category_game_mappings

2.2 수집 가능한 데이터 카테고리

오버레이를 통해 수집 가능한 데이터는 5개 카테고리로 분류된다.

2.2.1 시청자 상호작용 데이터

- **채팅**: 시청자 닉네임, 메시지 내용, 채팅 시간, 역할(구독자/VIP 등)
- **후원**: 후원자 닉네임, 후원 금액(KRW 정규화), 후원 메시지, 후원 종류(별풍선/치즈 등)
- **구독/팔로우**: 구독·팔로우 시청자, 구독 개월 수, 등급

2.2.2 방송 상태 데이터

- 방송 시작/종료 시점 및 지속 시간
- 크로스 플랫폼 실시간 시청자 수 (분단위 스냅샷)
- 방송 카테고리 및 카테고리 변경 이력 (세그먼트)
- 방송 제목 변경 추적

2.2.3 시청자-스트리머 관계 데이터

- 시청자별 채팅 횟수, 후원 횟수, 후원 총액 누적
- 시청자 첫 방문/최근 방문 추적
- 카테고리별 참여도 분석

2.2.4 게임/카테고리 트렌드 데이터

- 플랫폼별 카테고리 시청자·스트리머 수 시계열
- 크로스 플랫폼 통합 게임 카탈로그
- 카테고리 간 시청자 이동 패턴

2.2.5 세션 데이터

- 시청자 입퇴장 시점 및 체류 시간
- 분단위 채팅 통계 (채팅률 변화 추적)

제 3 장

데이터 수집 전략 — 왜 오버레이인가

3.1 오버레이의 전략적 위치

이를 꼭 플랫폼에서만 가져와야 하는가? 스트리머들이 쓰는 도구들에 주목했다.

오버레이 시스템은 인터넷 방송에서 필수적으로 쓰는 기능이다. 채팅을 화면에 올리고, 후원의 영상과 이미지를 보이고, 효과음을 낸다. 모든 스트리머는 이미 오버레이를 사용하고 있으며, 이 오버레이를 통해 **모든 이벤트가 지나간다**.

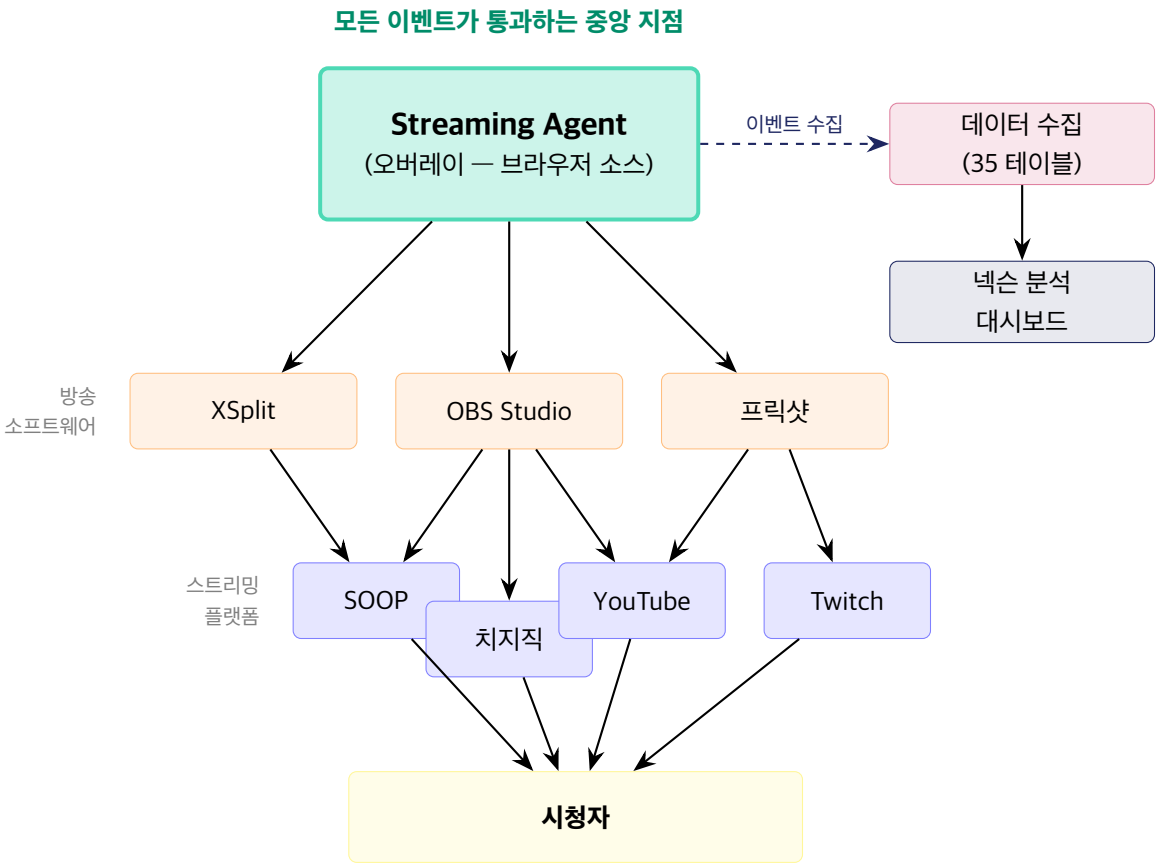

그림 3.1: 오버레이의 전략적 위치 — 방송 송출 플로우에서의 데이터 수집 지점

오버레이가 데이터 수집에 가장 이상적인 지점인 이유:

특성	의미
스트리머 워크플로우의 중앙	방송 시작 전 반드시 설정하는 필수 도구
플랫폼 종속 없음	스트리머가 직접 설치, 플랫폼 허락 불필요
항상 켜져 있음	방송 전체 시간 동안 실행
모든 이벤트가 통과	채팅, 후원, 구독, 팔로우 전부 처리

"데이터가 플랫폼에 있는 게 아니라, 사실은 스트리머의 화면 위를 다 지나가고 있었다."

3.2 플랫폼 독립적 데이터 파이프라인

Streaming Agent의 핵심 기술은 **이벤트 정규화(Event Normalization)**이다. 각 플랫폼은 서로 다른 이벤트 형식과 통화 체계를 사용하지만, 이를 하나의 통합 스키마(UnifiedEvent)로 변환한다.

표 3.1: 플랫폼별 후원 정규화 매핑

플랫폼	원본 타입	환산 방식	정규화 결과
SOOP	별풍선	1개 = 100 KRW	amount (KRW)
SOOP	애드벌룬	직접 KRW	amount (KRW)
SOOP	영상풍선	직접 KRW	amount (KRW)
치지직	치즈	직접 KRW	amount (KRW)
YouTube	Super Chat	USD $\rightarrow$ KRW	amount (KRW)
Twitch	Bits	1 bit = 14 KRW	amount (KRW)

3.3 경쟁사 분석 — 위플랩

위플랩(Weflab)은 국내 스트리머 오버레이 시장의 사실상 표준이다.

표 3.2: 위플랩 vs Streaming Agent 비교

항목	위플랩	Streaming Agent	비고
직원 수	5~15명	넥슨 지원	지속 투자 가능
오버레이 기능	완성도 높음	100% 대체 목표	10종 구현 완료
데이터 소유 구조	없음	**35개 테이블**	핵심 차별점
분석 대시보드	없음	**구현 완료**	스트리머 가치
사업 연계	불가	**넥슨 연동**	전략적 가치
BM 안정성	취약	**넥슨 백업**	대형 스트리머 신뢰

위플랩은 기능적 완성도가 높지만, BM이 확고하지 못해 소규모 인원이 운영하는 서비스에 대형 스트리머들마저 기대고 있는 상황이다. 이것은 기회이다.

현재 시장에는 “게임사 관점에서 스트리밍 데이터를 수집·정규화·활용하기 위한 서비스”가 존재하지 않는다.

제 4 장

서비스 구축 전략

4.1 Phase 1 — 스트리머 확보

4.1.1 전략 개요

위플랩보다 더 좋거나 최소 동일한 툴을 제공하여 시장을 장악하고, 스트리밍 데이터를 독점 확보하는 것이 1차 목표이다.

절대 원칙:

1. 넥슨 편향적으로 생각하지 않는다. 목표는 **많은 스트리머가 사용하게 하는 것**이 1순위.
2. 넥슨 가입 유도나 넥슨이라는 이름이 써있으면, 넥슨 게임 송출 스트리머 외에는 절대 안 쓴다.
3. 모든 게임사를 연결해주는 기능도 넣어야 한다.
4. 주기적인 업데이트가 필요하다.

4.1.2 스트리머 전환 전략

스트리머는 설정 바꾸는 것을 귀찮아 한다. 현재 기준 우리 것을 쓰게 하는 방법은 두 가지이다:

1. **쉽게 갈아타게 하기**: 위플랩 설정을 불러오는 마이그레이션 도구 제공
2. **우리만의 특별한 기능**: 이걸 써야만 이득을 보게 만들기

4.1.3 핵심 전략 4가지

넥슨 IP를 활용한 TTS/이펙트

- 넥슨 게임 캐릭터 TTS 보이스 (메이플스토리 세렌, 던전앤파이터 세리아 등)
- 다이내믹 모션 이펙트: 정적 GIF를 넘어 방송 화면 전체와 상호작용하는 모션 그래픽
- 시청자 이모티콘이 화면에 비처럼 내리거나 터지는 새로운 형태의 참여

스트리머용 분석 대시보드

어떤 게임, 어떤 장면에서 시청자 반응이 폭발했는지, 어떤 콘텐츠가 구독 전환율이 높았는지 등을 분석하여 데이터 기반 성장 전략 지원.

설정 템플릿 마켓플레이스

스트리머뿐 아니라 팬들도 템플릿을 생성할 수 있는 공간. 무료 디자인 공유 및 다운로드.

스트리머 파트너십 활용

- 넥슨 파트너 스트리머 / 광고 방송 송출 시 Streaming Agent 사용 조건 부여
- 신입/소규모 스트리머 발굴 및 지원 프로그램

4.1.4 구현 완료 오버레이 (10종)

표 4.1: Streaming Agent 오버레이 목록

#	오버레이	주요 기능
1	채팅창 (Chat)	26+ 테마, 애니메이션, 역할별 색상, 필터링, 내장 위젯(공지/타이머/시청자수)
2	후원 알림 (Alert)	TTS, 시그니처 알림, 금액별 커스텀, 룰렛 연동
3	후원 자막 (Subtitle)	MVP, 랭킹, 최근 목록, 이미지 후원 표시
4	목표치 (Goal)	바/원형/하트/별/반원 그래프, 자동 증가, 완료 이펙트
5	전광판 (Ticker)	스크롤 메시지, 권한 제한, 명령어 지원
6	룰렛 (Roulette)	세그먼트 커스텀, 트리거 조건, 스핀 애니메이션
7	이모지 (Emoji)	Float/Explode/Rain/Bounce 애니메이션
8	투표 (Voting)	최대 10개 옵션, 시간 제한, 실시간 결과
9	엔딩 크레딧 (Credits)	후원자/채팅 참여자 자동 수집, 배경음악
10	광고 (Ad)	Banner/Popup/Corner 슬롯, 실시간 성과 측정

4.2 Phase 2 — 넥슨 독점 기능

> **주의:** Phase 2는 현재 실행 대상이 아니다. 스트리머 시장의 **최소 80% 이상**이 Streaming Agent로 전환된 시점에서 비로소 검토를 시작하는 장기 비전이다. Phase 1의 중립적 포지셔닝이 완전히 확립되기 전에 넥슨 독점 기능을 도입하면 오히려 사용자 이탈을 초래할 수 있다.

충분한 시장 지배력이 확보된 이후, 넥슨 게임의 유저 유입 및 전환을 직접 유도하는 기능을 검토한다:

- **게임-방송 실시간 연동**: 게임 내 이벤트가 방송 오버레이에 실시간 반영
- **스트림 드롭스**: 방송 시청 시 게임 내 보상 제공 (시청자 유입 극대화)
- **인터랙티브 광고**: 오버레이 내 광고 슬롯을 통한 새로운 광고 인벤토리

이 기능들은 스트리머들이 이미 Streaming Agent에 의존하고 있는 상태에서 "추가 혜택"으로 제공되어야 의미가 있다. 시기상조에 도입하면 "넥슨 홍보 도구"라는 인식이 형성되어 서비스 전체의 중립성을 훼손한다.

4.3 중립성 유지 전략

표 4.2: 중립성 유지를 위한 핵심 원칙

원칙	실행 방안
넥슨 비노출	Phase 1에서는 넥슨 브랜딩을 서비스에 노출하지 않음
전 게임사 지원	넥슨뿐 아니라 모든 게임사의 카테고리·방송 데이터 수집
중립적 마켓플레이스	게임사 무관하게 모든 디자인 템플릿 허용
점진적 독점화	충분한 사용자 확보 후 넥슨 IP 기능을 “추가 혜택”으로 제공

제 5 장

기대효과 및 활용 시나리오

5.1 데이터의 사업적 가치

5.1.1 마케팅 및 사업개발

표 5.1: 마케팅 관점의 데이터 활용

활용	설명
인플루언서 발굴	표면적인 팔로워 수가 아닌, 각 게임에 대한 채팅 참여율과 긍정 반응률을 기반으로 잠재력 있는 스트리머를 경쟁사보다 먼저 발굴
캠페인 성과 측정	특정 스트리머의 방송 이후 채팅에서 우리 게임 언급량 변화, 긍정/부정 키워드를 정량적으로 분석
정밀 타겟 그룹 식별	특정 스트리머의 시청자들이 어떤 게임 성향인지 분석하여 타겟팅

5.1.2 게임 개발 및 운영

표 5.2: 게임 개발/운영 관점의 데이터 활용

활용	설명
유저 피드백	실시간 채팅에서 나오는 필터링되지 않은 반응을 즉각 수집
콘텐츠 매력도	스트리머와 시청자가 어떤 보스, 어떤 맵에서 가장 오래 머무는지 분석
경쟁 게임 분석	타 게임 카테고리의 시청자·스트리머 트렌드를 실시간 모니터링

5.2 구체적 활용 시나리오

5.2.1 시나리오 1: 진짜 인플루언서 발굴

데이터 소스: viewer_engagement + events + persons

팔로워 수가 아닌 실질적 참여도 기반으로 스트리머를 평가한다:

```
SELECT p.nickname, p.platform,
```

```
        SUM(ve.chat_count) as total_chats,
        SUM(ve.donation_count) as total_donations,
        SUM(ve.total_donation_amount) as total_amount,
        COUNT(DISTINCT ve.person_id) as unique_viewers
FROM persons p
JOIN viewer_engagement ve ON ve.broadcaster_person_id = p.id
WHERE ve.last_seen_at > datetime('now', '-30 days')
GROUP BY p.id
ORDER BY unique_viewers DESC, total_chats DESC;
```

비즈니스 인사이트: 이 쿼리를 통해 "팔로워는 적지만 실제 시청자 참여도가 높은 스트리머"를 발견할 수 있다. 기존 마케팅에서는 팔로워 수 중심으로 파트너십을 결정했으나, 이 데이터를 통해 **실질적 영향력** 기반의 의사결정이 가능하다.

5.2.2 시나리오 2: 캠페인 ROI 측정

데이터 소스: broadcast_segments + events + viewer_snapshots

특정 게임 캠페인 기간 동안의 시청자 반응 변화를 측정한다:

```
SELECT bs.category_name,
       COUNT(CASE WHEN e.event_type = 'chat' THEN 1 END) as chat_count,
       COUNT(CASE WHEN e.event_type = 'donation' THEN 1 END) as donation_count,
       AVG(vs.viewer_count) as avg_viewers
FROM broadcast_segments bs
JOIN events e ON e.broadcast_id = bs.broadcast_id
  AND e.event_timestamp BETWEEN bs.segment_started_at AND bs.segment_ended_at
LEFT JOIN viewer_snapshots vs ON vs.broadcast_id = bs.broadcast_id
WHERE bs.category_name LIKE '%MapleStory%'
  AND bs.segment_started_at > '2026-01-01'
GROUP BY bs.category_name;
```

비즈니스 인사이트: 캠페인 집행 전후 채팅량·후원량·시청자 수의 변화를 정량적으로 비교할 수 있다. "A 스트리머에게 1천만 원을 투자했을 때 시청자 참여가 얼마나 증가했는가"에 대한 구체적 답변이 가능해진다.

5.2.3 시나리오 3: 실시간 게임 트렌드 분석

데이터 소스: platform_categories + unified_games + category_stats

플랫폼 간 게임 인기도를 통합 비교한다. 예를 들어 "메이플스토리가 SOOP에서는 상위 5위이지만 치지직에서는 10위권 밖"이라는 인사이트를 실시간으로 제공한다.

5.2.4 시나리오 4: 시청자 세그먼트 분석

데이터 소스: user_sessions + viewer_engagement

시청자의 체류 시간, 참여 패턴, 이동 경로를 분석하여 타겟 마케팅에 활용한다. "넥슨 게임 방송을 자주 시청하는 시청자가 주로 보는 다른 게임은 무엇인가"를 파악할 수 있다.

제 6 장

사업 모델 분석

6.1 서비스 포지셔닝

Streaming Agent는 **영리 목적의 상용 서비스가 아니다.** 핵심 목표는 스트리밍 생태계의 데이터를 넥슨의 전략 자산으로 확보하는 것이며, 서비스 자체는 스트리머에게 최대한 무료로 제공하여 사용자 기반을 빠르게 확보하는 데 집중한다.

표 6.1: 서비스 운영 모델

구분	과금 여부	설명
오버레이 기능 (10종)	무료	채팅, 알림, 룰렛, 투표 등 모든 오버레이 기능을 무료 제공. 위플랩 대비 기능적 우위 확보가 목적
넥슨 IP 이펙트/TTS	무료	넥슨 캐릭터 TTS, 다이내믹 모션 이펙트 등 넥슨 IP 활용 콘텐츠를 무료 독점 제공하여 차별화
마켓플레이스	무료	디자인 템플릿 공유/다운로드를 무료로 운영. 크리에이터와 스트리머 간 커뮤니티 활성화
분석 대시보드	무료	스트리머에게 시청자 분석, 콘텐츠 성과 데이터를 무료 제공하여 서비스 충성도 확보

모든 기능을 무료로 제공함으로써 위플랩에서의 이탈 장벽을 최소화하고, 스트리머가 자발적으로 Streaming Agent를 선택하도록 유도한다. 넥슨의 IP와 기술력이 투입된 프리미엄 콘텐츠조차 무료로 제공한다는 점이 시장에서의 결정적 차별화 요소다.

6.2 수익 모델: 광고 인벤토리

현재 검토 중인 유일한 수익 모델은 **오버레이 내 광고 인벤토리**다.

6.2.1 광고의 구조적 차별점

오버레이에 표시되는 광고는 기존 웹 배너 광고와 근본적으로 다르다:

- **방송 화면 위 직접 노출**: 시청자가 반드시 보게 되는 위치에 광고가 송출됨

- **실시간 컨텍스트**: 방송 카테고리, 시청자 반응, 게임 종류에 따라 타겟팅 가능
- **높은 주목도**: 시청자가 능동적으로 시청 중인 콘텐츠 위에 노출되므로 기존 디스플레이 광고 대비 주목도가 높음
- **스트리머 수익 분배**: 스트리머와 수익을 분배하여 자발적 참여 유도 (예: 70:30 비율)

6.2.2 광고 슬롯 유형

표 6.2: 광고 슬롯 유형

유형	설명
Banner	방송 화면 상/하단에 배너 형태로 노출
Popup	특정 이벤트(후원, 구독 등) 발생 시 팝업 형태로 노출
Corner	화면 모서리에 소형 광고 상시 노출

6.2.3 광고 과금 모델

CPM(1,000회 노출당 과금) 또는 CPC(클릭당 과금) 방식을 검토 중이다. 오버레이 특성상 노출이 보장되므로 CPM 모델이 적합할 가능성이 높다.

6.3 시장 환경

표 6.3: 국내 스트리밍 오버레이 시장 현황

항목	현황
주요 플랫폼	SOOP(구 아프리카TV), 네이버 치지직
활성 스트리머 규모	수만 명 (양 플랫폼 합산 추정)
기존 오버레이 서비스	위플랩 (사실상 독점, 직원 5~15명, 추정 매출 3~10억 원)
경쟁 구도	경쟁자 부재로 시장 성장 정체. 게임사 관점의 오버레이 서비스 전무
진입 기회	넥슨의 IP·자본력·파트너십을 활용하면 빠른 시장 점유 가능

위플랩이 독점하고 있는 시장에 넥슨이 무료 + 고품질로 진입하면, 스트리머 입장에서 전환하지 않을 이유가 없다. 데이터 확보를 위한 사용자 기반 구축이 최우선이며, 광고 수익화는 충분한 사용자가 확보된 이후에 본격적으로 추진한다.

6.4 비용 구조

표 6.4: 연간 비용 구조 추정

항목	연간 추정	상세
개발 인력	4~6억 원	프론트엔드 2명, 백엔드 2명, 데이터 1명, PM 1명
인프라	1~2억 원	클라우드 서버, DB, CDN, 스토리지, 모니터링
운영	0.5~1억 원	고객 지원, QA, 커뮤니티 관리
마케팅	1~2억 원	스트리머 파트너십, 프로모션, 이벤트
합계	**6.5~11억 원**	

이 비용은 넥슨의 전략적 투자로 집행된다. 서비스 자체의 독립 수익으로 운영비를 충당하는 것이 1차 목표가 아니라, **데이터 확보를 통한 넥슨 전체 사업의 의사결정 품질 향상**이 본질적 투자 회수 방식이다.

제 7 장

리스크 및 대응 전략

7.1 사업 리스크 분석

표 7.1: 사업 리스크 분석 및 대응 전략

리스크	심각도	대응 방안
시장 저항 및 중립성 훼손	높음	Phase 1에서는 철저히 중립적 툴로 포지셔닝. 넥슨 관련 기능은 부가적 기능으로 명분 확보
스트리머 모객	높음	위플랩 마이그레이션 도구 제공, 마켓플레이스로 트렌디한 템플릿 자동 생성
플랫폼 정책 변경	중간	오버레이 자체가 플랫폼 독립적이므로 구조적 리스크 낮음. WebSocket 연결 방식은 모니터링
데이터 보안/개인정보	중간	개인정보보호 규정 설계, 익명 데이터 수집, 활용 목적 투명 고지. 최고 수준의 보안 체계
광고 수익 모델 불확실성	중간	광고 수익화는 충분한 사용자 확보 후 검토. 서비스 자체 수익보다 데이터 전략 자산 확보가 본질적 투자 가치
경쟁사 대응	낮음	위플랩은 소규모 팀으로 넥슨 IP 및 데이터 인프라 대응 불가. 글로벌 서비스는 한국 시장 특성상 진입 장벽 높음

7.2 리스크 완화 전략

1. **중립성 유지**: 출시 후 최소 1년간은 “넥슨”이라는 이름을 서비스 전면에 노출하지 않는다. 모든 게임사의 방송을 동등하게 지원하여 시장 신뢰를 확보한다.
2. **완전 무료 전략**: 모든 기능을 무료로 제공하여 진입 장벽을 제거한다. 광고 수익화는 사용자 기반이 충분히 확보된 이후 검토한다.
3. **데이터 투명성**: 수집하는 데이터의 범위와 활용 목적을 명확히 고지하고, 스트리머가 데이터 수집을 거부할 수 있는 옵션을 제공한다.

4. **지속적 혁신**: 마켓플레이스와 커뮤니티를 통해 사용자 피드백을 빠르게 반영하고, 위플랩 대비 기능적 우위를 유지한다.

제 8 장

실행 로드맵

8.1 단계별 실행 계획

표 8.1: 단계별 실행 계획 요약

단계	목표 시점	핵심 마일스톤
현재 (완료)	2026 Q1	10종 오버레이, 35개 DB 테이블, 50+ API, 실시간 파이프라인 구축
Phase 1 전반	2026 Q2~Q3	넥슨 IP TTS 출시, 위플랩 마이그레이션 도구, 마켓플레이스 오픈
Phase 1 후반	2026 Q4~2027 Q1	분석 대시보드 고도화, 게임별 성과 리포트, 광고 시스템 검증
Phase 2	2027 Q2~	넥슨 독점 기능 (스트림 드롭스, 인게임 연동), 광고 인벤토리 본격 운영

8.2 핵심 KPI

표 8.2: 단계별 핵심 KPI

KPI	Phase 1 목표	Phase 2 목표
위플랩 대비 점유율	30%	70%
일일 수집 이벤트	100만 건	500만 건
월 활성 시청자(MAU)	50만 명	200만 명
데이터 활용 부서 수	3개 부서	10개 부서
광고 시스템	검증 완료	본격 운영

8.3 투자 대비 기대 효과

넥슨이 이 프로젝트에 투자해야 하는 이유를 정리한다:

1. **데이터 독점**: 경쟁사가 접근할 수 없는 실시간 스트리밍 데이터를 넥슨만의 전략 자산으로 확보

2. **마케팅 ROI 측정**: 감각이 아닌 데이터 기반의 스트리밍 마케팅 의사결정 가능
3. **생태계 지배력**: 스트리밍 인프라를 장악하여 넥슨 게임의 방송 노출·유입 극대화
4. **방어적 가치**: 경쟁 게임사가 유사 서비스를 구축하기 전에 시장 선점
5. **광고 인프라 확보**: 사용자 기반 확보 후 오버레이 광고라는 새로운 광고 채널을 운영할 수 있는 기반 마련

제 II 편

기술 명세

제 9 장

시스템 아키텍처

9.1 전체 시스템 구조

표 9.1: 기술 스택

영역	기술
Frontend	React 19, Vite 7, React Router DOM 7, Lucide React, Recharts
Backend	Express 5, Socket.io 4, SQLite3
인증	JWT (jsonwebtoken), bcrypt, OAuth (SOOP, Naver, Google, Twitch)
스타일	CSS Custom Properties, Glass-morphism
로깅	Pino
캐싱	Redis (선택)

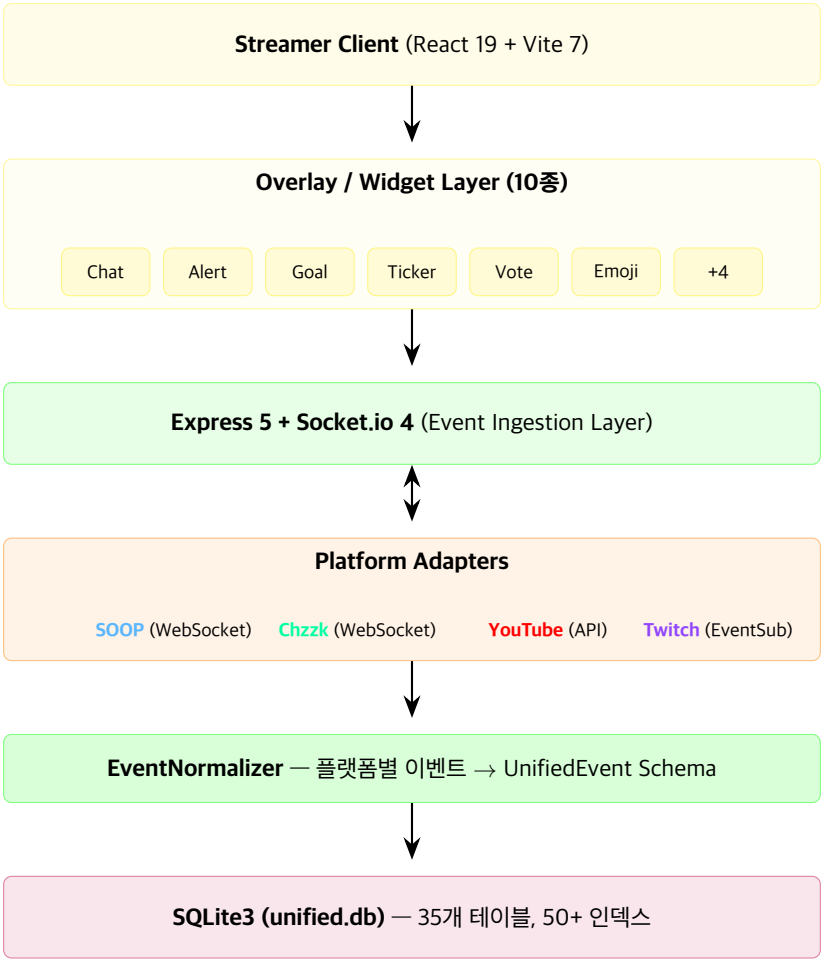

그림 9.1: 시스템 레이어 아키텍처

9.2 플랫폼 어댑터

각 플랫폼은 서로 다른 프로토콜과 이벤트 형식을 사용한다. Streaming Agent는 BaseAdapter 패턴을 통해 플랫폼별 차이를 추상화한다.

표 9.2: 플랫폼별 어댑터 특성

플랫폼	프로토콜	이벤트 타입	특이사항
SOOP	WebSocket	chat, donation (5종), follow	별풍선/애드벌룬/영상풍선/미션/스티커 구분
치지직	WebSocket	chat, donation, subscribe	치즈 단일 통화
YouTube	REST API	chat, super_chat, member	다국가 통화 환산 필요
Twitch	EventSub	chat, bits, subscribe, follow	Bits 환산 (1bit = 14KRW)

9.3 이벤트 정규화 파이프라인

모든 플랫폼 이벤트는 다음의 UnifiedEvent 스키마로 정규화된다:

Listing 9.1: UnifiedEvent 스키마

```
{
  id: UUID,                      // 고유이벤트 ID
  type: 'chat' | 'donation' | 'subscribe' | 'follow',
  platform: 'chzzk' | 'soop' | 'youtube' | 'twitch',
  sender: {
    id: string,                  // 플랫폼사용자 ID
    nickname: string,            // 표시닉네임
    role: 'streamer' | 'manager' | 'vip' | 'fan' | 'regular',
    profileImage: string         // 프로필이미지 URL
  },
  content: {
    message: string,             // 메시지내용
    amount: number,              // KRW 환산금액
    originalAmount: number,      // 원본금액
    currency: string,            // 원본통화
    donationType: string         // 플랫폼별후원타입
  },
  metadata: {
    timestamp: ISO8601,          // 이벤트발생시각
    channelId: string,           // 채널 ID
    broadcastId: string,         // 방송 ID
    categoryId: string,          // 카테고리 ID
    categoryName: string         // 카테고리이름
  }
}
```

제 10 장

데이터 스키마 상세

본 장은 Streaming Agent가 수집·저장하는 모든 데이터의 구조를 상세히 기술한다. 총 35개 테이블이 6개 그룹으로 분류된다.

10.1 스키마 개요

그림 10.1: 데이터베이스 스키마 그룹 구조 (35개 테이블)

표 10.1: 테이블 그룹 요약

그룹	테이블 수	목적
코어 스트리밍	9	이벤트, 방송, 시청자, 세그먼트 등 핵심 데이터
카테고리/게임	5	크로스 플랫폼 카테고리 통합 및 트렌드
오버레이 설정	10	10종 오버레이 위젯의 설정 데이터
사용자/인증	4	계정, JWT 토큰, 세션 관리
광고 시스템	4	광고 슬롯, 캠페인, 노출/클릭, 정산
마켓플레이스	3	디자인 크리에이터, 템플릿, 리뷰

10.2 코어 스트리밍 데이터

10.2.1 persons — 인물 통합 테이블

스트리머와 시청자를 하나의 테이블에서 관리한다. channel_id의 유무로 방송인과 시청자를 구분한다.

표 10.2: persons 테이블 (15 컬럼)

컬럼	타입	제약	설명
id	INTEGER	PK, AUTO	고유 식별자
platform	TEXT	NOT NULL	soop | chzzk | twitch | youtube
platform_user_id	TEXT	NOT NULL	플랫폼 내 사용자 ID
nickname	TEXT		표시 닉네임
profile_image_url	TEXT		프로필 이미지 URL
channel_id	TEXT		채널 ID (NULL = 시청자)
channel_description	TEXT		채널 소개
follower_count	INTEGER	DEFAULT 0	팔로워 수
subscriber_count	INTEGER	DEFAULT 0	구독자 수
total_broadcast_minutes	INTEGER	DEFAULT 0	총 방송 시간 (분)
last_broadcast_at	DATETIME		마지막 방송 시각
first_seen_at	DATETIME	DEFAULT NOW	최초 발견 시각
last_seen_at	DATETIME	DEFAULT NOW	최근 활동 시각
created_at	DATETIME	DEFAULT NOW	생성 시각
updated_at	DATETIME	DEFAULT NOW	수정 시각

UNIQUE(platform, platform_user_id)

10.2.2 events — 이벤트 통합 테이블

채팅, 후원, 구독, 팔로우 등 모든 이벤트를 하나의 테이블에 기록한다. UUID 기반 PK로 중복 방지.

표 10.3: events 테이블 (16 컬럼)

컬럼	타입	제약	설명
id	TEXT	PK	UUID v4
event_type	TEXT	NOT NULL	chat | donation | subscribe | follow | view
platform	TEXT	NOT NULL	soop | chzzk | twitch | youtube
actor_person_id	INTEGER	FK→persons	이벤트 발생 주체
actor_nickname	TEXT		비정규화된 닉네임 (빠른 조회)
actor_role	TEXT		streamer | manager | vip | fan | regular
target_person_id	INTEGER	FK→persons	이벤트 대상 (스트리머)
target_channel_id	TEXT	NOT NULL	대상 채널 ID
broadcast_id	INTEGER	FK→broadcast	방송 ID
message	TEXT		채팅/후원 메시지
amount	INTEGER		KRW 환산 금액
original_amount	INTEGER		원본 금액
currency	TEXT		원본 통화
donation_type	TEXT		별풍선/치즈/bits 등
event_timestamp	DATETIME	NOT NULL	이벤트 발생 시각
ingested_at	DATETIME	DEFAULT NOW	수집 시각

10.2.3 broadcasts — 방송 테이블

표 10.4: broadcasts 테이블 (18 컬럼)

컬럼	타입	제약	설명
id	INTEGER	PK, AUTO	고유 식별자
platform	TEXT	NOT NULL	플랫폼
channel_id	TEXT	NOT NULL	채널 ID
broadcast_id	TEXT	NOT NULL	플랫폼 방송 ID
broadcaster_person_id	INTEGER	FK→persons	방송인
title	TEXT		방송 제목
thumbnail_url	TEXT		썸네일
current_viewer_count	INTEGER	DEFAULT 0	현재 시청자 수
peak_viewer_count	INTEGER	DEFAULT 0	최대 시청자 수
avg_viewer_count	INTEGER	DEFAULT 0	평균 시청자 수
viewer_sum	INTEGER	DEFAULT 0	시청자 합산 (평균 계산용)
snapshot_count	INTEGER	DEFAULT 0	스냅샷 횟수
is_live	INTEGER	DEFAULT 1	라이브 여부

컬럼	타입	제약	설명
started_at	DATETIME		방송 시작
ended_at	DATETIME		방송 종료
duration_minutes	INTEGER		방송 시간 (분)
recorded_at	DATETIME	DEFAULT NOW	기록 시각
updated_at	DATETIME	DEFAULT NOW	수정 시각

UNIQUE(platform, channel_id, broadcast_id)

10.2.4 broadcast_segments — 카테고리 변경 세그먼트

방송 중 카테고리(게임)가 변경될 때마다 새로운 세그먼트가 생성된다. 이를 통해 "어떤 게임을 얼마나 했는지"를 정확히 추적할 수 있다.

표 10.5: broadcast_segments 테이블 (10 컬럼)

컬럼	타입	제약	설명
id	INTEGER	PK, AUTO	고유 식별자
broadcast_id	INTEGER	FK, CASCADE	방송 ID
platform	TEXT	NOT NULL	플랫폼
channel_id	TEXT	NOT NULL	채널 ID
category_id	TEXT		카테고리 ID
category_name	TEXT		카테고리 이름
segment_started_at	DATETIME	NOT NULL	세그먼트 시작
segment_ended_at	DATETIME		세그먼트 종료
peak_viewer_count	INTEGER	DEFAULT 0	구간 최대 시청자
avg_viewer_count	INTEGER	DEFAULT 0	구간 평균 시청자

10.2.5 viewer_engagement — 시청자-방송인 관계

특정 시청자가 특정 방송인의 채널에서 보인 참여도를 누적 기록한다.

표 10.6: viewer_engagement 테이블 (12 컬럼)

컬럼	타입	제약	설명
id	INTEGER	PK, AUTO	고유 식별자
person_id	INTEGER	FK→persons	시청자
platform	TEXT	NOT NULL	플랫폼
channel_id	TEXT	NOT NULL	채널 ID
broadcaster_person_id	INTEGER	FK→persons	방송인

category_id	TEXT		카테고리
chat_count	INTEGER	DEFAULT 0	채팅 횟수 누적
donation_count	INTEGER	DEFAULT 0	후원 횟수 누적
total_donation_amount	INTEGER	DEFAULT 0	후원 총액 (KRW)
first_seen_at	DATETIME	DEFAULT NOW	첫 참여 시각
last_seen_at	DATETIME	DEFAULT NOW	마지막 참여
updated_at	DATETIME	DEFAULT NOW	수정 시각

UNIQUE(person_id, channel_id, platform, category_id)

10.2.6 viewer_snapshots — 시청자 수 시계열

분단위로 시청자 수와 채팅률을 스냅샷으로 기록한다.

표 10.7: viewer_snapshots 테이블 (9 컬럼)

컬럼	타입	제약	설명
id	INTEGER	PK, AUTO	고유 식별자
platform	TEXT	NOT NULL	플랫폼
channel_id	TEXT	NOT NULL	채널 ID
broadcast_id	INTEGER	FK	방송 ID
segment_id	INTEGER	FK	세그먼트 ID
viewer_count	INTEGER	NOT NULL	시청자 수
chat_rate_per_minute	INTEGER		분당 채팅 수
snapshot_at	DATETIME	NOT NULL	스냅샷 시각
ingested_at	DATETIME	DEFAULT NOW	수집 시각

10.2.7 chat_stats — 분단위 채팅 통계

유저별 분단위 채팅 횟수를 집계한다.

표 10.8: chat_stats 테이블 (8 컬럼)

컬럼	타입	제약	설명
id	BIGINT	PK, AUTO	고유 식별자
platform	VARCHAR(20)	NOT NULL	플랫폼
channel_id	VARCHAR(255)	NOT NULL	채널 ID
user_id	VARCHAR(255)	NOT NULL	유저 ID
user_nickname	VARCHAR(255)		유저 닉네임
minute_timestamp	TIMESTAMP	NOT NULL	분단위 타임스탬프

chat_count	INTEGER	DEFAULT 1	해당 분 채팅 수
created_at	TIMESTAMP	DEFAULT NOW	생성 시각

UNIQUE(platform, channel_id, user_id, minute_timestamp)

10.3 게임/카테고리 카탈로그

크로스 플랫폼 카테고리 통합은 “타 게임, 타 카테고리 방송 데이터 수집”이라는 핵심 목표를 달성하기 위한 핵심 구조이다.

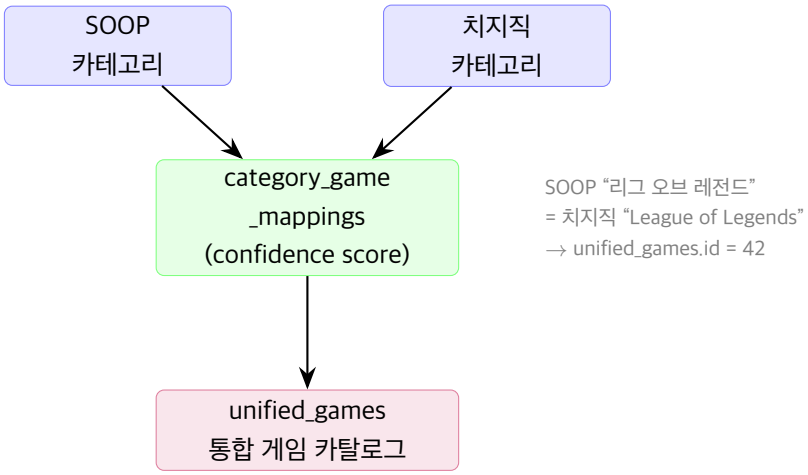

그림 10.2: 크로스 플랫폼 카테고리 통합 흐름

표 10.9: platform_categories 테이블 (11 컬럼)

컬럼	타입	제약	설명
id	INTEGER	PK, AUTO	고유 식별자
platform	TEXT	NOT NULL	플랫폼
platform_category_id	TEXT	NOT NULL	플랫폼 내 카테고리 ID
platform_category_name	TEXT	NOT NULL	카테고리 이름
category_type	TEXT		GAME | IRL | MUSIC
thumbnail_url	TEXT		썸네일
viewer_count	INTEGER	DEFAULT 0	현재 시청자 수
streamer_count	INTEGER	DEFAULT 0	현재 스트리머 수
is_active	INTEGER	DEFAULT 1	활성 여부
first_seen_at	DATETIME	DEFAULT NOW	최초 발견
last_seen_at	DATETIME	DEFAULT NOW	최근 활동

UNIQUE(platform, platform_category_id)

표 10.10: unified_games 테이블 (10 컬럼)

컬럼	타입	제약	설명
id	INTEGER	PK, AUTO	고유 식별자
name	TEXT	NOT NULL	게임명 (영문)
name_kr	TEXT		게임명 (한국어)
genre	TEXT		장르 (영문)
genre_kr	TEXT		장르 (한국어)
developer	TEXT		개발사
release_date	TEXT		출시일
description	TEXT		설명
image_url	TEXT		대표 이미지
is_verified	INTEGER	DEFAULT 0	검증 여부

표 10.11: category_game_mappings 테이블 (7 컬럼)

컬럼	타입	제약	설명
id	INTEGER	PK, AUTO	고유 식별자
unified_game_id	INTEGER	FK, CASCADE	통합 게임 ID
platform	TEXT	NOT NULL	플랫폼
platform_category_id	TEXT	NOT NULL	플랫폼 카테고리 ID
platform_category_name	TEXT		플랫폼 카테고리 이름
confidence	REAL	DEFAULT 1.0	매핑 신뢰도 (0~1)
is_manual	INTEGER	DEFAULT 0	수동 매핑 여부

UNIQUE(platform, platform_category_id)

10.4 오버레이 설정 테이블

오버레이 설정은 settings (글로벌) 및 user_settings (유저별) 테이블에 JSON 형태로 저장된다. 특수 기능은 전용 테이블을 사용한다.

표 10.12: 오버레이 설정 관련 테이블

테이블	컬럼 수	용도
settings	2	글로벌 설정 (key-value, JSON)
user_settings	5	유저별 설정 (key-value, JSON)
roulette_wheels	9	룰렛 세그먼트, 트리거 조건
signature_sounds	9	금액별 커스텀 알림음/이미지
emoji_settings	8	이모지 세트, 애니메이션 스타일
voting_polls	7	투표 제목, 옵션, 상태
poll_votes	5	투표 기록
ending_credits	10	크레딧 섹션, 배경음악, 자동 수집
chat_bots	4	봇 설정
bot_commands	5	봇 명령어

10.5 사용자/인증 및 광고/마켓플레이스

10.5.1 사용자 시스템

표 10.13: users 테이블 (12 컬럼)

컬럼	타입	제약	설명
id	INTEGER	PK, AUTO	고유 식별자
email	TEXT	UNIQUE	이메일
password_hash	TEXT		bcrypt 해시
display_name	TEXT	NOT NULL	표시 이름
avatar_url	TEXT		아바타 URL
role	TEXT	DEFAULT 'user'	user | creator | advertiser | admin
oauth_provider	TEXT		OAuth 제공자
oauth_id	TEXT		OAuth ID
overlay_hash	TEXT	UNIQUE	오버레이 고유 해시 (16자 hex)
channel_id	TEXT		연결된 채널 ID
platform	TEXT		주 플랫폼
created_at	DATETIME	DEFAULT NOW	가입 시각

10.5.2 광고 시스템

표 10.14: 광고 시스템 테이블

테이블	컬럼 수	용도
ad_slots	9	스트리머 광고 슬롯 (위치, 크기, 타입)
ad_campaigns	15	광고주 캠페인 (예산, 기간, 타겟, 상태)
ad_impressions	6	노출/클릭 이벤트 기록
ad_settlements	7	스트리머 정산 (기간별 노출·클릭·수익)

10.5.3 마켓플레이스

표 10.15: 마켓플레이스 테이블

테이블	컬럼 수	용도
creators	8	크리에이터 프로필 (다운로드 수, 평점, 인증)
designs	12	디자인 템플릿 (카테고리, 승인 워크플로우)
design_reviews	6	리뷰 (1~5점 평점, 텍스트)

10.6 실시간 이벤트 구조 (Socket.io)

표 10.16: Socket.io 이벤트 목록

이벤트	방향	설명
join-overlay	Client → Server	오버레이 룸 참가 (overlay:{hash})
leave-overlay	Client → Server	오버레이 룸 퇴장
new-event	Server → Client	채팅/후원/구독/팔로우 이벤트
settings-update	Client → Server	설정 변경 요청
settings-updated	Server → Client	설정 변경 알림
roulette-spin	Bidirectional	룰렛 스핀
emoji-reaction	Server → Client	단일 이모지 반응
emoji-burst	Server → Client	다중 이모지 반응
poll-start/vote/end	Bidirectional	투표 라이프사이클
credits-start/stop	Client → Server	엔딩 크레딧 제어
ad-slots-updated	Server → Client	광고 슬롯 변경 알림

new-event 페이로드:

```
{
  type: "donation",           // chat | donation | subscribe | follow
  sender: "testuser",         // 발신자닉네임
  amount: 10000,              // KRW 환산금액
  message: "test message",    // 메시지
  platform: "soop",           // 플랫폼
  timestamp: "2026□01□30T12:00:00Z",
  id: "uuid□v4"               // 고유 ID
}
```

제 11 장

API 명세

11.1 REST API 엔드포인트

표 11.1: API 그룹별 엔드포인트 요약

그룹	엔드포인트 수	주요 기능
인증	6	회원가입, 로그인, OAuth, 프로필, 토큰 갱신
설정	6	글로벌/유저 설정 CRUD, 오버레이 설정
광고	10	슬롯 CRUD, 캠페인 CRUD, 노출/클릭, 정산
관리자	5	통계, 스트리머 목록, 플랫폼 통계, 모니터링
모니터링	12	방송/인물/이벤트/세그먼트/카테고리 조회
카테고리	8	카테고리 CRUD, 통합 게임, 매핑, 트렌드
플랫폼	8	SOOP/치지직 연결·해제·상태, 채널 정보
헬스체크	3	health, ready, detailed
유틸리티	2	이벤트 시뮬레이션, 이벤트 조회
합계	**∼60**	

11.1.1 인증 API

Method	Endpoint	설명
POST	/api/auth/register	회원가입 (email, password, displayName)
POST	/api/auth/login	로그인
GET	/api/auth/me	현재 사용자 정보
PUT	/api/auth/profile	프로필 업데이트
GET	/api/auth/:provider	OAuth 로그인 (google, naver, twitch, soop)
POST	/api/auth/refresh	토큰 갱신

11.1.2 모니터링 API

넥슨 내부 대시보드를 위한 핵심 API:

Method	Endpoint	설명
GET	/api/monitor/stats	전체 통계 (방송·시청자·후원 집계)
GET	/api/monitor/stats/timeseries	시계열 통계 (시간별 추이)
GET	/api/monitor/stats/nexon	넥슨 게임 전용 통계
GET	/api/monitor/broadcasts	방송 목록 (라이브/종료, 페이지네이션)
GET	/api/monitor/persons	인물 목록 (방송인/시청자, 검색)
GET	/api/monitor/persons/:id	인물 상세 (방송이력, 참여도, 이벤트)
GET	/api/monitor/engagement	시청자 참여도 순위
GET	/api/monitor/events	이벤트 목록 (타입별 필터)
GET	/api/monitor/categories	카테고리 목록
GET	/api/monitor/segments	방송 세그먼트 목록
GET	/api/monitor/snapshots	시청자 스냅샷 목록
GET	/api/monitor/schema	데이터베이스 스키마 정보

11.2 Socket.io 실시간 이벤트

Socket.io는 오버레이 실시간 동기화의 핵심이다. 각 오버레이 인스턴스는 overlay:{userHash} 룸에 참가하여, 해당 스트리머의 이벤트만 수신한다.

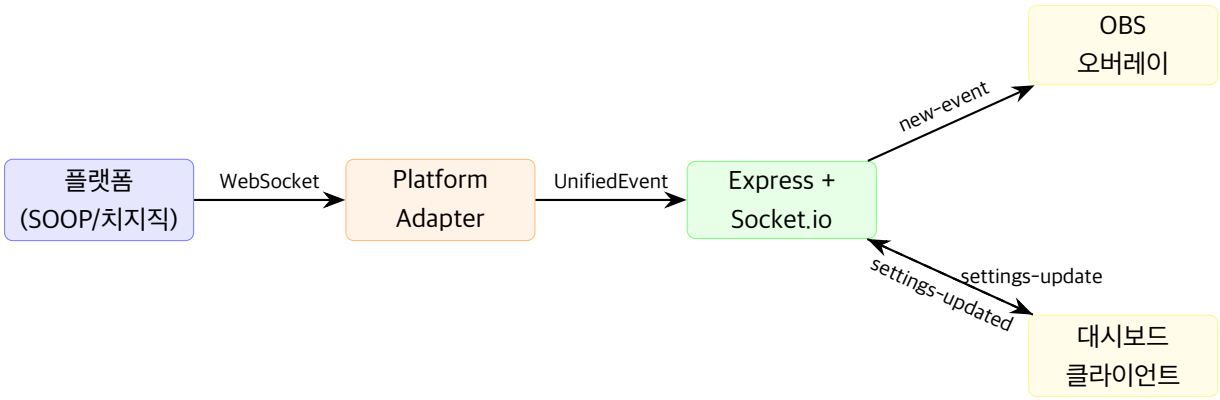

그림 11.1: Socket.io 이벤트 흐름

제 12 장

개발 로드맵 및 기술 과제

12.1 개발 로드맵

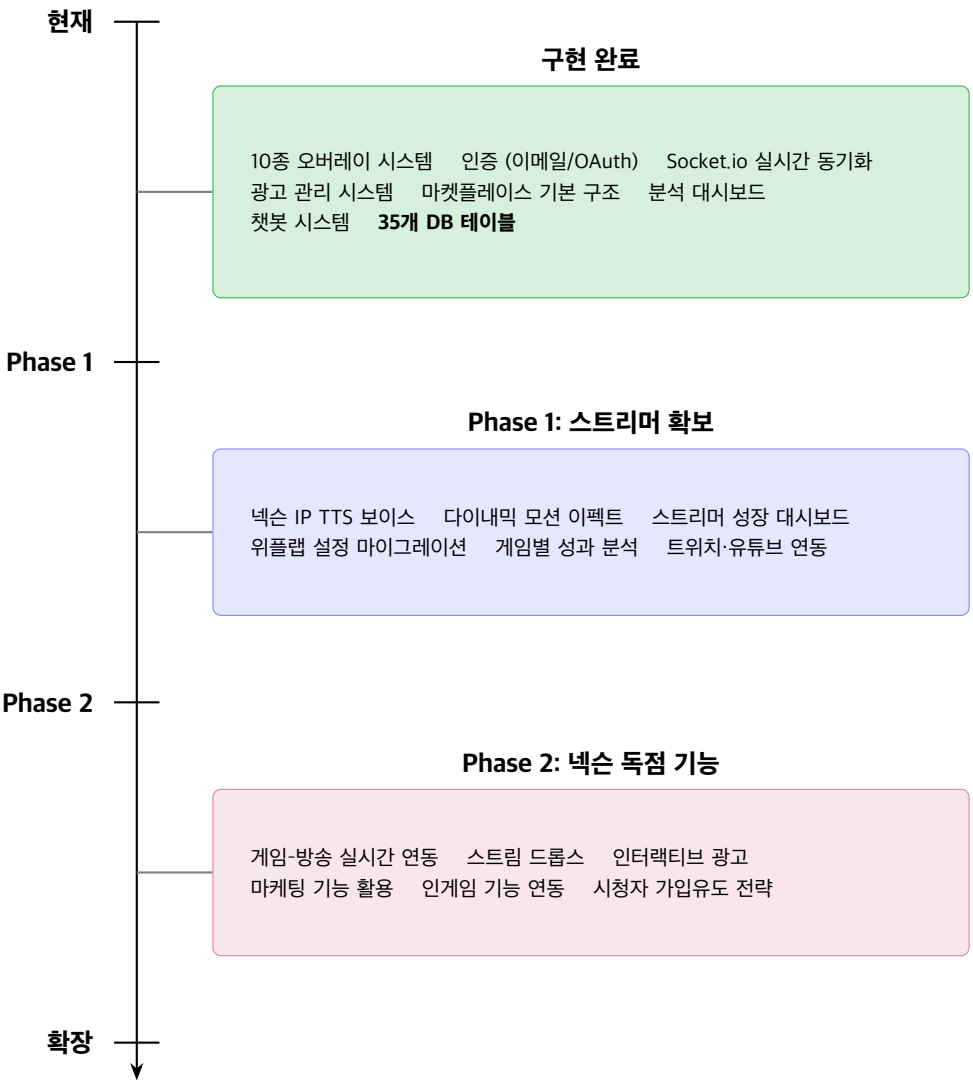

그림 12.1: 개발 로드맵

12.2 기술 과제

표 12.1: 핵심 기술 과제 및 대응 방안

과제	우선순위	대응 방안
확장성	높음	SQLite에서 PostgreSQL로 마이그레이션, 수평 확장 아키텍처 도입. 이벤트 처리량 10만 건/분 목표
데이터 파이프라인 고도화	높음	메시지 큐(Kafka/Redis Streams) 도입, 이벤트 소싱 패턴 적용, 데이터 유실 방지
실시간 처리 성능	중간	WebSocket 연결 풀 최적화, Socket.io 클러스터링, CDN 기반 오버레이 배포
데이터 품질	중간	정규화 파이프라인의 지속적 검증, 이상치 탐지 알고리즘, A/B 테스트 인프라
보안 강화	높음	API Rate Limiting 고도화, DDoS 방어, 민감 데이터 암호화, 접근 제어 세분화
모니터링/관측성	중간	분산 트레이싱, 실시간 알림, 성능 대시보드, 로그 중앙화(ELK/Loki)

12.3 추후 확장 플랜

1. **글로벌 플랫폼 연동**: Twitch EventSub, YouTube Data API v3 완전 통합
2. **마이크로서비스 전환**: 모놀리식 Express 서버를 기능별 마이크로서비스로 분리
3. **ML 파이프라인**: 시청자 행동 예측, 감성 분석, 이상 탐지 자동화
4. **인게임 SDK**: 게임 내 이벤트 $\leftrightarrow$ 방송 오버레이 실시간 연결을 위한 SDK 개발

결론

본 프로젝트는 단순한 오버레이 툴 개발이 아니라, 스트리밍 생태계의 데이터 주권을 확보하기 위한 전략적 사업이다.

> "플랫폼이 데이터를 안 주니까 만든 게 아니라,
> 우리가 스트리밍 생태계의 중심이 되기 위해 만든 것이다."

Part I: 사업 전략에서 살펴본 바와 같이, Streaming Agent는 스트리밍 데이터를 넥슨의 전략 자산으로 전환하는 데이터 파이프라인이다. 모든 기능을 무료로 제공하여 위플랩 대비 압도적인 가치를 제공하고, 넥슨 IP를 활용한 차별화 콘텐츠로 스트리머 기반을 빠르게 확보한다. 광고 인벤토리를 통한 수익화를 검토 중이나, 서비스 자체의 영리보다 데이터 확보를 통한 넥슨 전체 사업의 의사결정 품질 향상이 핵심 투자 가치다.

Part II: 기술 명세에서 제시한 시스템 아키텍처는 이 사업 전략을 뒷받침하는 기술적 기반이다. 플랫폼 어댑터 패턴을 통한 멀티 플랫폼 지원, 35개 테이블로 구성된 정규화 스키마, 그리고 RESTful API + Socket.io 실시간 통신 체계가 안정적인 데이터 수집과 서비스 운영을 보장한다.

이 구조의 핵심 장점:

항목	기존 방식	Streaming Agent
플랫폼 허락	필수	불필요
정책 리스크	높음	거의 없음
데이터 범위	API 한계	이벤트 무제한
속도	느림 (배치)	실시간
지속 가능성	정책 변경 시 중단	독립적
수익 모델	플랫폼 종속	광고 + 데이터 전략 자산

이건 우회가 아니라 구조적 승리다.

부록 가

API 엔드포인트 상세

가.1 인증 API

가.1.1 POST /api/auth/register

Request Body:

```
{
  "email": "user@example.com",
  "password": "password123",
  "displayName": "nickname"
}
```

Response (200):

```
{
  "token": "eyJhbGc...",
  "user": {
    "id": 1,
    "email": "user@example.com",
    "displayName": "nickname",
    "role": "user",
    "overlayHash": "abc123def456"
  }
}
```

가.2 설정 API

가.2.1 GET /api/settings/:key

Parameters: key = chat | alert | goal | subtitle | ticker | roulette | emoji | voting | credits

Response:

```
{
  "key": "chat",
  "value": "{\"theme\":\"default\",\"fontSize\":28,...}"
}
```

가.2.2 POST /api/settings

Request Body:

```
{
  "key": "chat",
  "value": {
    "theme": "default",
    "fontSize": 28,
    "alignment": "left"
  }
}
```

저장 시 Socket.io로 settings-updated 이벤트 브로드캐스트.

가.3 모니터링 API

가.3.1 GET /api/monitor/stats

넥슨 내부 대시보드의 핵심 API:

```
{
  "liveBroadcasts": 150,
  "totalViewers": 45000,
  "totalPersons": 12000,
  "totalDonations": 5600000,
  "platforms": {
    "soop": { "broadcasts": 80, "viewers": 25000 },
    "chzzk": { "broadcasts": 70, "viewers": 20000 }
  },
  "nexon": {
    "soop": { "broadcasts": 15, "viewers": 8000 },
    "chzzk": { "broadcasts": 12, "viewers": 6500 }
  }
}
```

가.4 에러 응답 형식

```
{
  "error": "Error message",
  "code": "ERROR_CODE"
}
```

코드	설명
400	Bad Request — 잘못된 요청
401	Unauthorized — 인증 필요
403	Forbidden — 권한 없음
404	Not Found — 리소스 없음
500	Internal Server Error

부록 나

데이터베이스 CREATE TABLE 전문

나.1 코어 스트리밍

Listing 나.1: persons

```
CREATE TABLE persons (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  platform TEXT NOT NULL CHECK(platform IN ('soop','chzzk','twitch','youtube')),
  platform_user_id TEXT NOT NULL,
  nickname TEXT,
  profile_image_url TEXT,
  channel_id TEXT,
  channel_description TEXT,
  follower_count INTEGER DEFAULT 0,
  subscriber_count INTEGER DEFAULT 0,
  total_broadcast_minutes INTEGER DEFAULT 0,
  last_broadcast_at DATETIME,
  first_seen_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  last_seen_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  UNIQUE(platform, platform_user_id)
);
```

Listing 나.2: events

```
CREATE TABLE events (
  id TEXT PRIMARY KEY,
  event_type TEXT NOT NULL CHECK(event_type IN
    ('chat','donation','subscribe','follow','view')),
  platform TEXT NOT NULL CHECK(platform IN ('soop','chzzk','twitch','youtube')),
  actor_person_id INTEGER REFERENCES persons(id),
  actor_nickname TEXT,
  actor_role TEXT CHECK(actor_role IN
    ('streamer','manager','vip','fan','regular','system')),
  target_person_id INTEGER REFERENCES persons(id),
  target_channel_id TEXT NOT NULL,
  broadcast_id INTEGER REFERENCES broadcasts(id),
  message TEXT,
  amount INTEGER,
  original_amount INTEGER,
  currency TEXT,
  donation_type TEXT,
  event_timestamp DATETIME NOT NULL,
```

```
  ingested_at DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

Listing 나.3: broadcasts

```
CREATE TABLE broadcasts (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  platform TEXT NOT NULL CHECK(platform IN ('soop','chzzk','twitch','youtube')),
  channel_id TEXT NOT NULL,
  broadcast_id TEXT NOT NULL,
  broadcaster_person_id INTEGER REFERENCES persons(id),
  title TEXT,
  thumbnail_url TEXT,
  current_viewer_count INTEGER DEFAULT 0,
  peak_viewer_count INTEGER DEFAULT 0,
  avg_viewer_count INTEGER DEFAULT 0,
  viewer_sum INTEGER DEFAULT 0,
  snapshot_count INTEGER DEFAULT 0,
  is_live INTEGER DEFAULT 1,
  started_at DATETIME,
  ended_at DATETIME,
  duration_minutes INTEGER,
  recorded_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  UNIQUE(platform, channel_id, broadcast_id)
);
```

나.2 사용자 시스템

Listing 나.4: users

```
CREATE TABLE users (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  email TEXT UNIQUE,
  password_hash TEXT,
  display_name TEXT NOT NULL,
  avatar_url TEXT,
  role TEXT DEFAULT 'user' CHECK(role IN ('user','creator','advertiser','admin')),
  oauth_provider TEXT,
  oauth_id TEXT,
  overlay_hash TEXT UNIQUE,
  channel_id TEXT,
  platform TEXT,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

부록 다

플랫폼별 이벤트 정규화 매핑

표 다.1: 플랫폼 원본 → UnifiedEvent 매핑

플랫폼	원본 필드	UnifiedEvent	변환	비고
SOOP	bj_id	sender.id	직접 매핑	
SOOP	user_nick	sender.nickname	직접 매핑	
SOOP	star_cnt	content.amount	× 100 KRW	별풍선
SOOP	ad_amount	content.amount	직접 KRW	애드벌룬
SOOP	grade	sender.role	등급 변환	1=streamer 등
치지직	uid	sender.id	직접 매핑	
치지직	nickname	sender.nickname	직접 매핑	
치지직	payAmount	content.amount	직접 KRW	치즈
치지직	profile.userRoleCode	sender.role	코드 변환	streamer/manager 등
YouTube	authorChannelId	sender.id	직접 매핑	
YouTube	displayName	sender.nickname	직접 매핑	
YouTube	amountMicros	content.amount	USD→KRW	Super Chat
YouTube	currency	content.currency	보존	다국가 통화
Twitch	user_id	sender.id	직접 매핑	
Twitch	user_name	sender.nickname	직접 매핑	
Twitch	bits	content.amount	× 14 KRW	Bits
Twitch	badge_info	sender.role	배지 변환	subscriber/vip 등