10장. 알림 시스템 설계

1단계 | 문제 이해 및 설계 범위 확정

하루에 백만 건 이상의 알림을 처리하는 확장성 높은 시스템을 구축하는 게 쉬운 과제는 아니다. 알림 시스템이 어떻게 구현되는지에 대한 깊은 이해가 필요한 작업이다. 따라서 적절한 질문을 통해 요구 사항이 무엇인지 지원자 스스로 알아내야 한다.

2단계 | 개략적 설계안 제시 및 동의 구하기

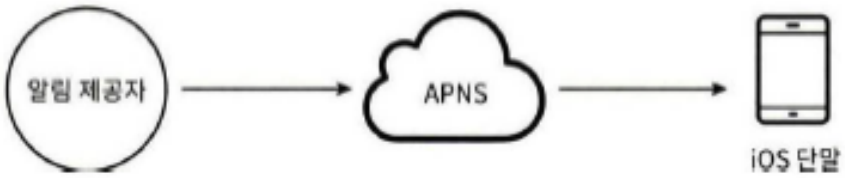
이번 절에서는 **IOS 푸시 알림**, **안드로이드 푸쉬 알림**, **SMS 메시지**, 그리고 **이메일을 지원하는 알림 시스템**의 개략적 설계안을 살펴볼 것이다. 다음과 같은 내용을 다룬다.

- 알림 유형별 지원 방안
- 연락처 정보 수집 절차
- 알림 전송 및 수신 절차

알림 유형별 지원 방안

우선 각각의 알림 매커니즘이 어떻게 동작하는지부터 알아보자.

IOS 푸시 알림



- IOS에서 푸시 알림을 보내기 위해서는 세 가지 컴포넌트가 필요
  - 알림 제공자(provider): 알림 요청을 만들어 애플 푸시 알림 서비스(APNS: Apple Push Notification Service)로 보내는 주체. 알림 요청을 만들려면 다음과 같은 데이터가 필요
    - 단말 토큰(device token): 알림 요청을 보내는 데 필요한 고유 식별자
    - 페이로드(payload): 알림 내용을 담은 JSON 딕셔너리

```
{
 "aps": {
  "alert": {
   "title": "Game Request",
   "body": "minu wants to play lol",
```

```
      "action_loc_key": "PLAY"
    },
    "badge": 5
    }
}
```

- APNS: 애플이 제공하는 원격 서비스. 푸시 알림을 iOS 장치로 보내는 역할을 담당
- iOS 단말: 푸시 알림을 수신하는 사용자 단말

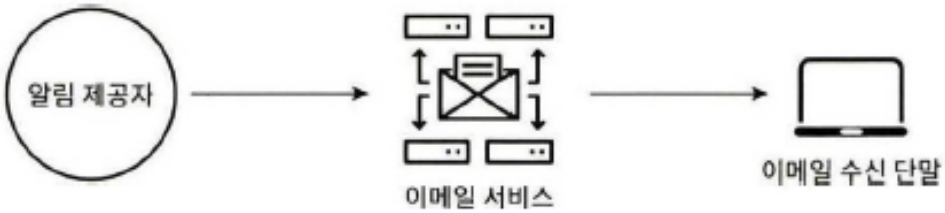
안드로이드 푸시 알림

- 안드로이드 푸시 알림도 iOS 푸시 알림과 비슷한 절차이나, APNS 대신 FCM(Firebase Cloud Messaging)을 사용

SMS 메시지

- SMS 메시지를 보낼 때는 보통 트윌리오(Twilio), 넥스모(Nexmo) 같은 제3 사업자의 서비스를 많이 이용(이런 서비스는 대부분 상용 서비스라 요금을 내야 함)

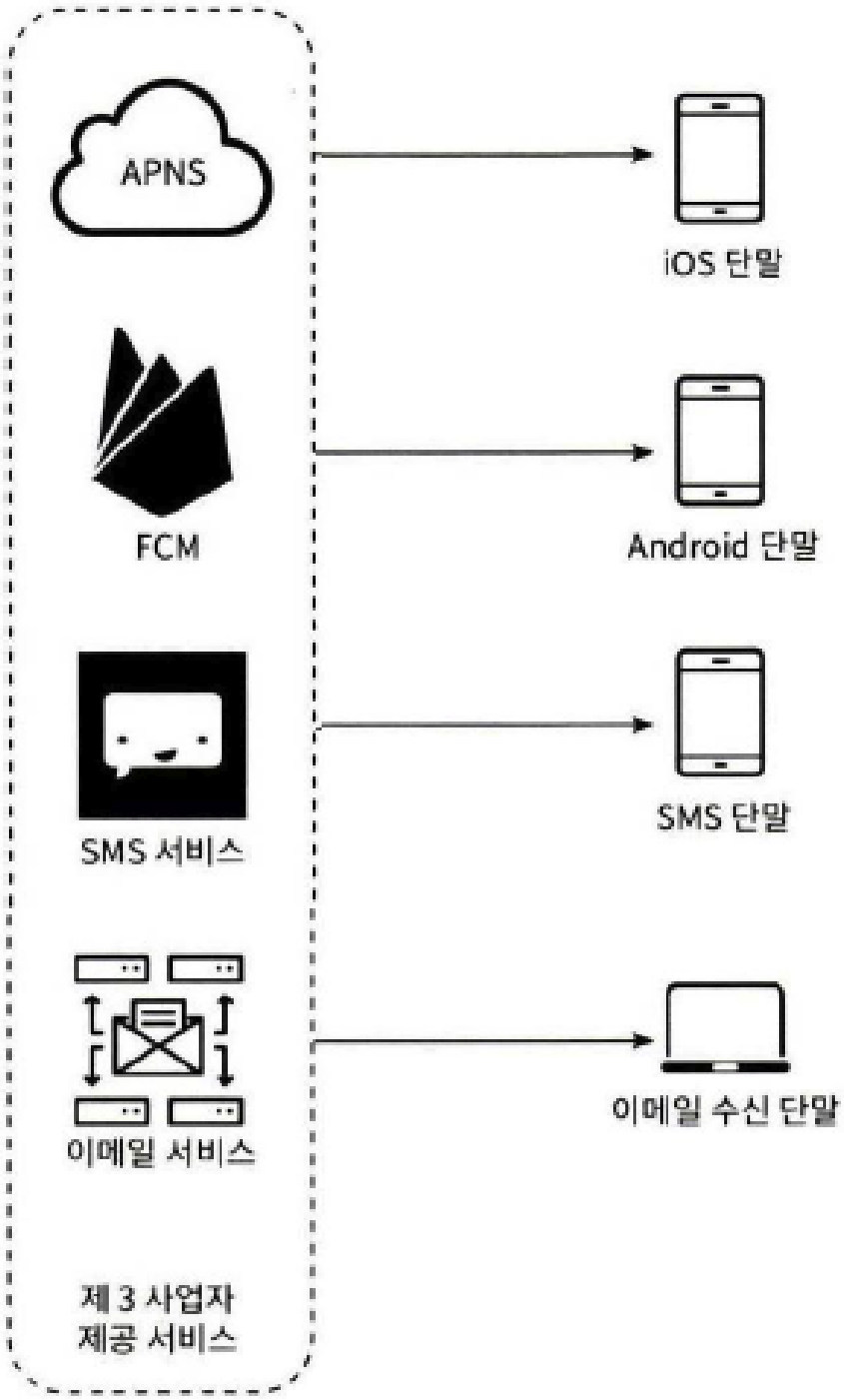
이메일



- 대부분의 회사는 고유 이메일 서버를 구축할 역량을 갖추고 있지만, 그럼에도 많은 회사가 상용 이메일 서비스를 이용

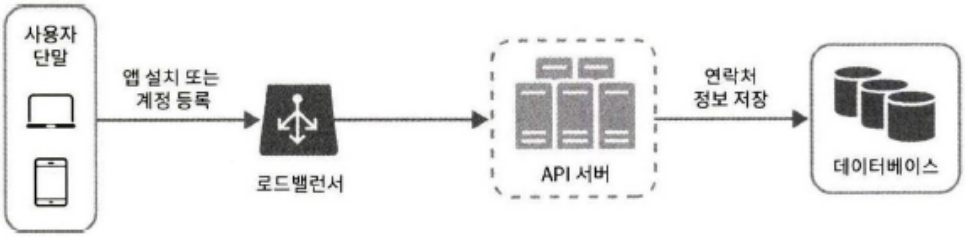
- 그중 유명한 서비스로 센드그리드, 메일침프가 있음. 전송 성공률도 높고, 데이터 분석 서비스도 제공

다음 그림은 지금까지 살펴본 알림 유형을 한 시스템으로 묶은 결과

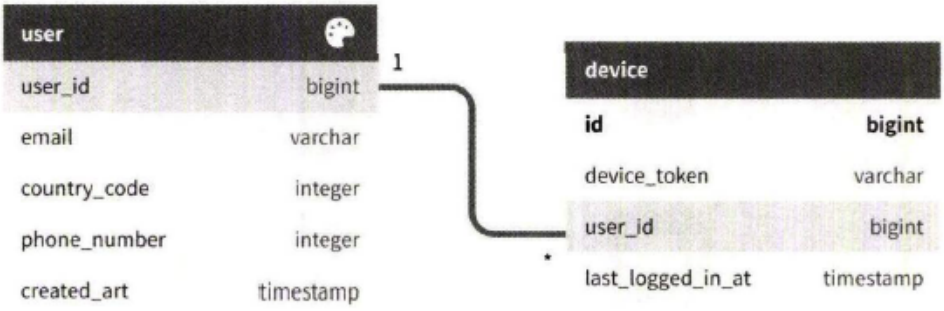


연락처 정보 수집 절차

알림을 보내려면 모바일 단말 토큰, 전화번호, 이메일 주소 등의 정보가 필요하다. 다음 그림과 같이 사용자가 앱을 설치하거나 처음으로 계정을 등록하면 API 서버는 해당 사용자의 정보를 수집하여 데이터베이스에 저장한다.

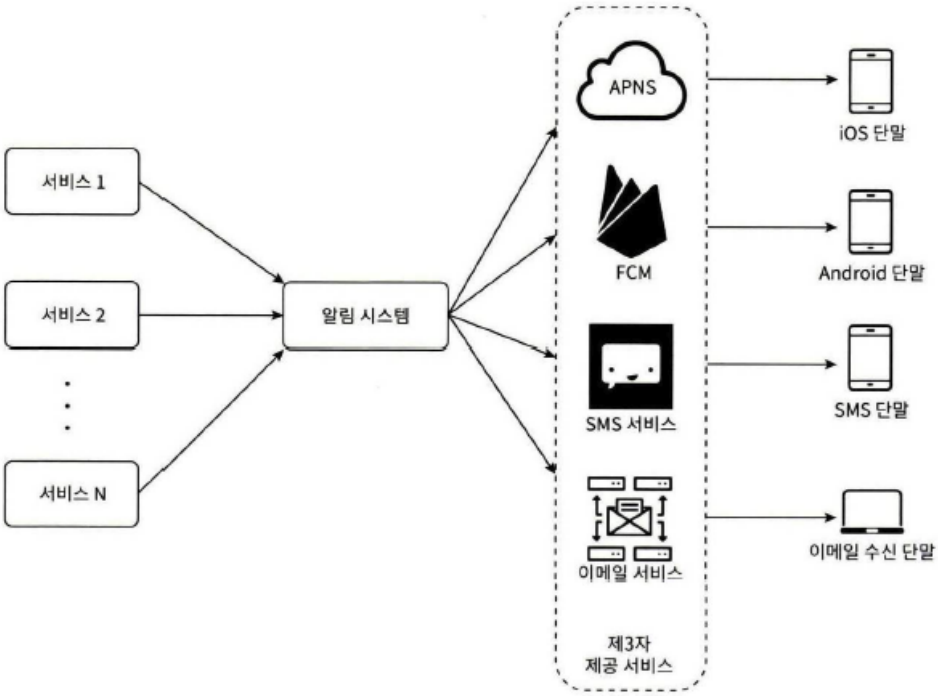


이 데이터베이스에 연락처 정보를 저장할 테이블 구조는 다음 그림과 같다. 필수 정보만 담은 개략적인 설계안으로서, 이메일 주소와 전화번호는 user 테이블에 저장하고, 단말 토큰은 device 테이블에 저장한다. 한 사용자가 여러 단말을 가질 수 있고, 알림은 모든 단말에 전송되어야 한다는 점을 고려하였다.



알림 전송 및 수신 절차

개략적 설계안 (초안)



- 1부터 N까지의 서비스

    - 이 서비스 각각은 마이크로서비스 일 수도 있고, 크론잡일 수도 있고, 분산 시스템 컴포넌트 일 수도 있다. 사용자에게 납기일을 알리고자 하는 과금 서비스, 배송 알림을 보내려는 쇼핑몰 웹사이트 등이 그 예다.
- 알림 시스템
    - 알림 시스템은 알림 전송/수신 처리의 핵심이다. 우선은 1개의 서버만 사용하는 시스템이라고 가정해 보자. 이 시스템은 서비스 1~N에 알림 전송을 위한 API를 제공해야 하고, 제3자 서비스에 전달할 알림 페이로드를 만들어 낼 수 있어야 한다.
- 제3자 서비스(third party services)
    - 이 서비스들은 사용자에게 알림을 실제로 전달하는 역할을 한다. 제3자 서비스와의 통합을 진행할 때 유의할 것은 확장성(extensibility)이다. 쉽게 새로운 서비스를 통합하거나 기존 서비스를 제거할 수 있어야 한다는 뜻이다.
    - 또 하나 고려해야 할 것은, 어떤 서비스는 다른 시장에서는 사용하지 못할 수도 있다는 것이다. 가령 FCM은 중국에서는 사용할 수 없다. 따라서 중국 시장에서는 제이푸시(Jpush), 푸시와이(PushY) 같은 서비스를 사용해야만 한다.
- iOS, AOS, SMS, 이메일 단말
    - 사용자는 자기 단말에서 알림을 수신한다.

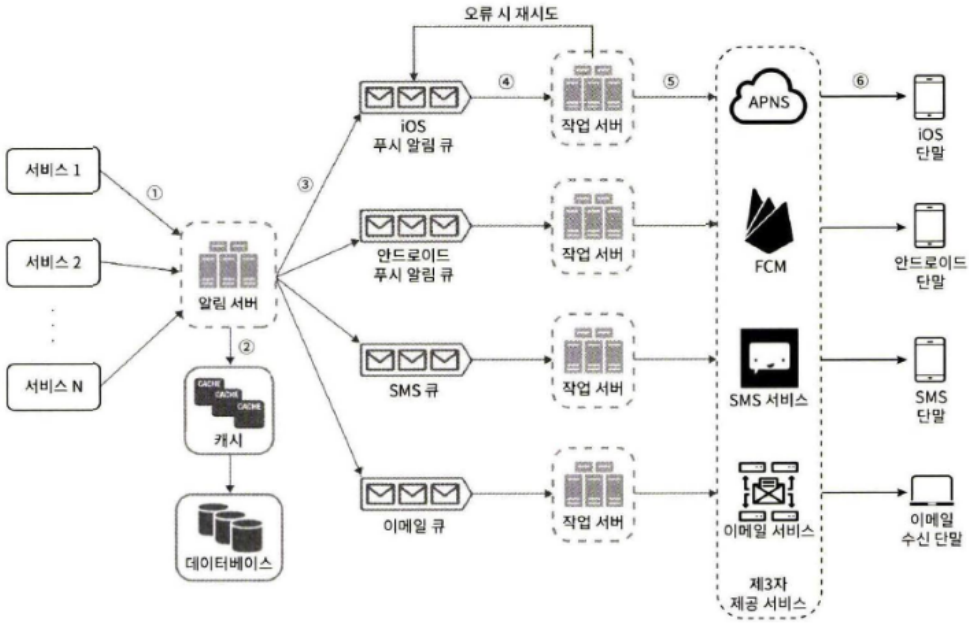
문제점

- SPOF(Single Point Of Failure)
    - 알림 서비스에 서버가 하나밖에 없다는 것은, 그 서버에 장애가 생기면 전체 서비스의 장애로 이어진다는 뜻이다.
- 규모 확장성
    - 한 대 서비스로 푸시 알림에 관계돈 모든 것을 처리하므로, 데이터베이스나 캐시 등 중요 컴포넌트의 규모를 개별적으로 늘릴 방법이 없다.
- 성능 병목
    - 알림을 처리하고 보내는 것은 자원을 많이 필요로 하는 작업일 수 있다. 예를 들어 HTML 페이지를 만들고 제3자 서비스의 응답을 기다리는 일은 시간이 많이 걸릴 가능성이 있는 작업이다. 따라서, 모든 것을 한 서버로 처리하면 사용자의 트래픽이 많이 몰리는 시간에는 시스템이 과부하 상태에 빠질 수 있다.

개략적 설계안 (개선된 버전)

개선 방향

- 데이터베이스와 캐시를 알림 시스템의 주 서버에서 분리
- 알림 서버를 증설하고 자동으로 수평적 규모 확장이 이루어질 수 있도록 함
- 메시지 큐를 이용해 시스템 컴포넌트 사이의 강한 결합을 끊음



- 1부터 N까지의 서비스
  - 알림 시스템 서버의 API를 통해 알림을 보낼 서비스들
- 알림 서버(notification server) - 다음 기능들을 제공
  - 알림 전송 API: 스팸 방지를 위해 보통 사내 서비스 또는 인증된 클라이언트만 사용 가능하다.
  - 알림 검증: 이메일 주소, 전화번호 등에 대한 기본적 검증을 수행한다.
  - 데이터베이스 또는 캐시 질의: 알림에 포함시킬 데이터를 가져오는 기능이다.
  - 알림 전송: 알림 데이터를 큐에 넣는다. 본 설계안의 경우 하나 이상의 메시지 큐를 사용하므로 알림을 병렬적으로 처리할 수 있다.
  - 다음은 이메일 형태의 알림을 보내는 데 사용하는 API 예제
    - `POST https://api.example.com/v/smms/send`
      - API 호출 시 전송할 데이터(body)의 사례

```
{
 "to": [
  {
   "user_id": 123456
  }
 ],
 "from": {
  "email": "from_address@example.com"
 },
 "subject": "Hello, WORLD!",
 "content": [
  {
   "type": "text/plain",
   "value": "Hello, World!"
  }
 ]
}
```

- 캐시(cache)
  - 사용자 정보, 단말 정보, 알림 템플릿 등을 캐시한다.
- 데이터베이스
  - 사용자, 알림, 설정 등 다양한 정보를 저장한다.
- 메시지 큐
  - 시스템 컴포넌트 간 의존성을 제거하기 위해 사용한다.
  - 다량의 알림이 전송되어야 하는 경우를 대비한 버퍼 역할도 한다.
  - 본 설계에서는 알림의 종류별로 별도 메시지 큐를 사용한다. 따라서 제3자 서비스 가운데 하나에 장애가 발생해도 다른 종류의 알림은 정상 동작한다.
- 작업 서버(workers)
  - 메시지 큐에서 전송할 알림을 꺼내어 제3자 서비스로 전달하는 역할을 담당하는 서버다.
- 제3자 서비스
- iOS, AOS, SMS, 이메일 단말

알림 전송 과정

1. API를 호출하여 알림 서버로 알림을 보낸다.
2. 알림 서버는 사용자 정보, 단말 토큰, 알림 설정 같은 메타데이터를 캐시나 데이터베이스에서 가져온다.
3. 알림 서버는 전송할 알림에 맞는 이벤트를 만들어서 해당 이벤트를 위한 큐에 넣는다. 가령 iOS 푸시 알림 이벤트는 iOS 푸시 알림 큐에 넣어야 한다.
4. 작업 서버는 메시지 큐에서 알림 이벤트를 꺼낸다.
5. 작업 서버는 알림을 제3자 서비스로 보낸다.
6. 제3자 서비스는 사용자 단말로 알림을 전송한다.

3단계 | 상세 설계

이 섹션에서는 다음 내용을 좀 더 자세히 알아본다.

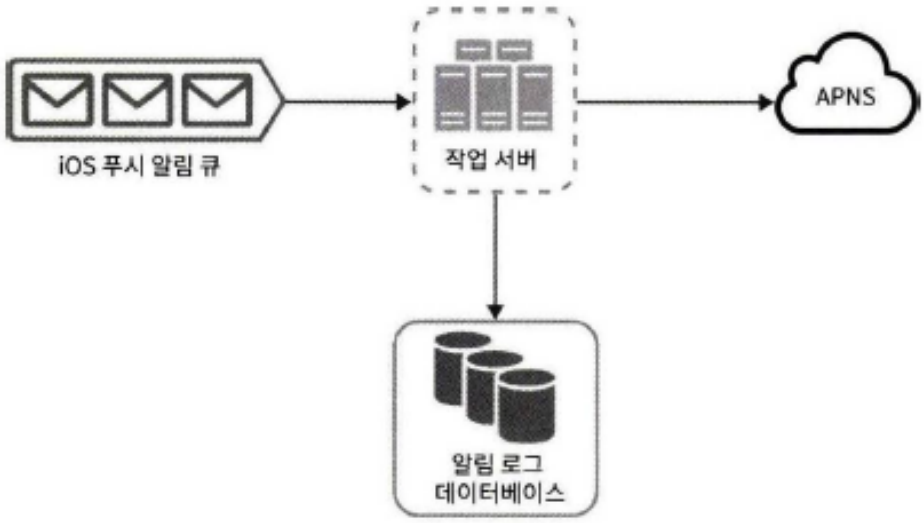
- 안정성(reliability)
- 추가로 필요한 컴포넌트 및 고려사항
  - 알림 템플릿, 알림 설정, 전송률 제한, 재시도 메커니즘, 보안, 큐에 보관된 알림에 대한 모니터링과 이벤트 추적 등이 이에 해당한다.
- 개선된 설계안

안정성

분산 환경에서 운영될 알림 시스템을 설계할 때는 안정성을 확보하기 위한 사항 몇 가지를 반드시 고려해야 한다.

데이터 손실 방지

알림 전송 시스템의 가장 중요한 요구사항 가운데 하나는 어떤 상황에서도 알림이 소실되면 안 된다는 것이다. 알림이 지연되거나 순서가 틀려도 괜찮지만, 사라지면 곤란하다. 이 요구사항을 만족하려면 알림 시스템은 알림 데이터를 데이터베이스 보관하고 재시도 메커니즘을 구현해야 한다. 다음 그림과 같이 알림 로그 데이터베이스를 유지하는 것이 한 가지 방법이다.



알림 중복 전송 방지

같은 알림이 여러 번 반복되는 것을 완전히 막는 것은 가능하지 않다. 대부분의 경우 딱 한 번만 전송되겠지만, 분산 시스템의 특성상 가끔은 같은 알림이 중복되어 전송되기도 할 것이다. 그 빈도를 줄이려면 중복을 탐지하는 메커니즘을 도입하고, 오류를 신중하게 처리해야 한다. 다음은 간단한 중복 방지 로직의 사례다.

- 보내야 할 알림이 도착하면 그 이벤트 ID를 검사하여 이전에 본 적이 있는 이벤트인지 살핀다. 중복된 이벤트라면 버리고, 그렇지 않으면 알림을 발송

추가로 필요한 컴포넌트 및 고려사항

지금까지 사용자 연락처 정보를 어떻게 수집하고, 알림은 어떻게 보내고 받을 것인지 살펴보았다. 그러나 알림 시스템은 이보다 훨씬 복잡하다. 지금부터는 알림 템플릿, 알림 설정, 이벤트 추적, 시스템 모니터링, 처리율 제한 등 알림 시스템 구현을 위해 필요한 추가 컴포넌트들에 대해 알아보자.

알림 템플릿

대형 알림 시스템은 하루에도 수백만 건 이상의 알림을 처리한다. 그런데 그 알림 메시지 대부분은 형식이 비슷하다. 알림 템플릿은 이런 유사성을 고려하여, 알림 메시지의 모든 부분을 처음부터 다시 만들 필요 없도록 해 준다. 알림 템플릿은 인자(parameter)나 스타일, 추적 링크(trackink link)를 조정하기만 하면 사전에 지정한 형식에 맞춰 알람을 만들어 내는 틀이다.

알림 설정

사용자는 이미 너무 많은 알림을 받고 있어서 쉽게 피곤함을 느낀다. 따라서 많은 웹사이트와 앱에서는 사용자가 알림 설정을 상세히 조정할 수 있도록 하고 있다. 이 정보는 알림 설정 테이블에 보관되며, 이 테이블에는 아마 다음과 같은 필드들이 필요할 것이다.

```
user_id     bigInt
channel     varchar     # 알림이 전송될 채널. 푸시 알림, 이메일, SMS 등
opt_in      boolean     # 해당 채널로 알림을 받을 것인지의 여부
```

이와 같은 설정을 도입한 뒤에는 특정 종류의 알림을 보내기 전에 반드시 해당 사용자가 해당 알림을 켜 두었는지 확인해야 한다.

전송률 제한

사용자에게 너무 많은 알림을 보내지 않도록 하는 한 가지 방법은, 한 사용자가 받을 수 있는 알림의 빈도를 제한하는 것이다. 이것이 중요한 이유는, 알림을 너무 많이 보내기 시작하면 사용자가 알림 기능을 아예 꺼 버릴 수도 있기 때문이다.

재시도 방법

제3자 서비스가 알림 전송에 실패하면, 해당 알림을 재시도 전용 큐에 넣는다. 같은 문제가 계속해서 발생하면 개발자에게 통지한다(alert).

푸시 알림과 보안

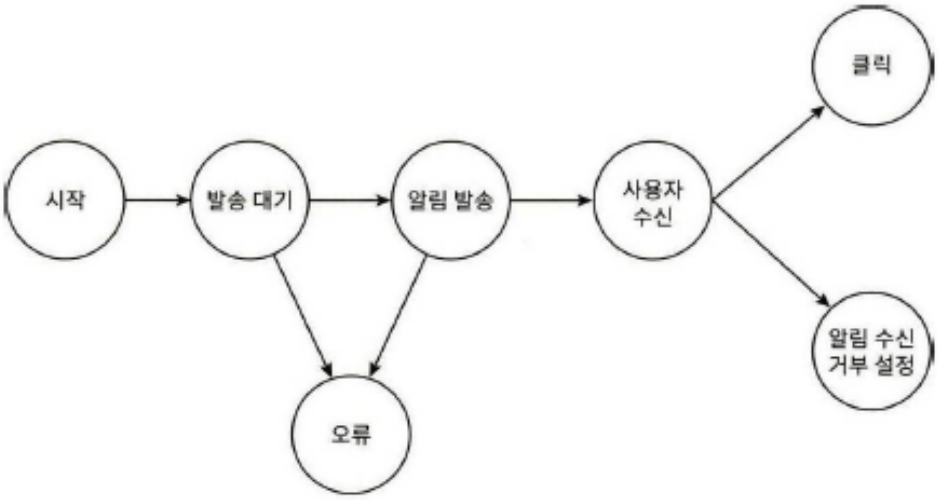
iOS와 안드로이드 앱의 경우, 알림 전송 API는 appKey와 appSecret을 사용하여 보안을 유지한다. 따라서 인증된, 혹은 승인된 클라이언트만 해당 API를 사용하여 알림을 보낼 수 있다.

큐 모니터링

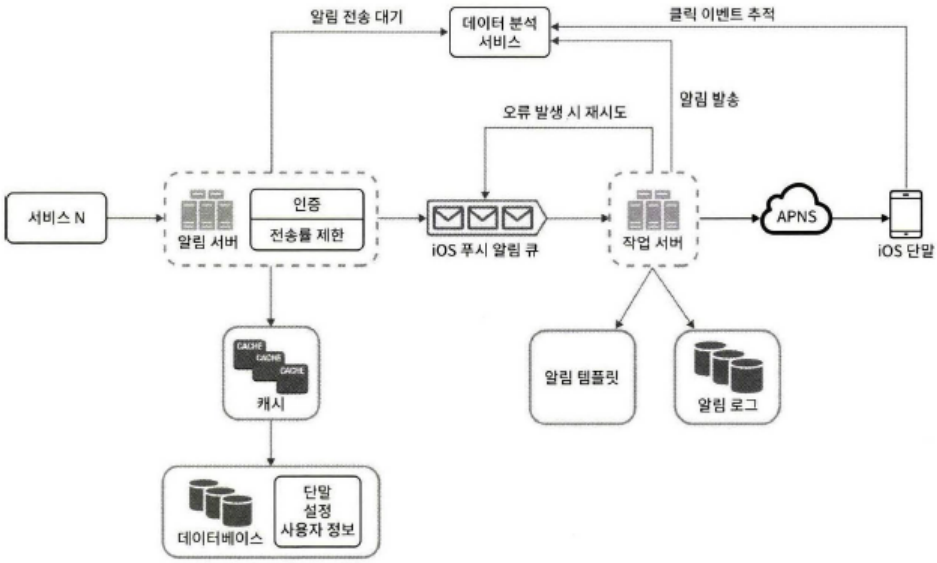
알림 시스템을 모니터링 할 때 중요한 메트릭 하나는 큐에 쌓인 알림의 개수이다. 이 수가 너무 크면 작업 서버들이 이벤트를 빠르게 처리하고 있지 못한다는 뜻으로, 작업 서버를 증설하는 게 바람직하다.

이벤트 추적

알림 확인율, 클릭율, 실제 앱 사용으로 이어지는 비율 같은 메트릭은 사용자를 이해하는데 중요하다. 데이터 분석 서비스는 보통 이벤트 추적 기능도 제공한다. 따라서 보통 알림 시스템을 만들면 데이터 분석 서비스와도 통합해야만 한다. 다음 그림은 데이터 분석 서비스를 통해 추적하게 될 알람 시스템 이벤트의 사례다.



수정된 설계안



이전의 설계안에서 개선된 부분은 아래와 같다.

- 알림 서버에 인증(authentication)과 전송률 제한 기능이 추가되었다.
- 전송 실패에 대응하기 위한 재시도 기능이 추가되었다. 전송에 실패한 알림은 다시 큐에 넣고 지정된 횟수만큼 재시도한다.
- 전송 템플릿을 사용하여 알림 생성 과정을 단순화하고 알림 내용의 일관성을 유지한다.
- 모니터링과 추적 시스템을 추가하여 시스템 상태를 확인하고 추후 시스템을 개선하기 쉽도록 하였다.

4단계 | 마무리

개략적 설계안과 더불어 각 컴포넌트의 구현 방법과 최적화 기법에 대해서도 심도 있게 알아보았다. 특히 아래 주제에 집중하였다.

- 안정성(reliablility)
  - 메세지 전송 실패율을 낮추기 위해 안정적인 재시도 메커니즘을 도입
- 보안
  - 인증된 클라이언트만이 알림을 보낼 수 있도록 appKey, appSecret 등의 메커니즘을 이용
- 이벤트 추적 및 모니터링
  - 알림이 만들어진 후 성공적으로 전송되기까지의 과정을 추적하고 시스템 상태를 모니터링 하기 위해 알림 전송의 각 단계마다 이벤트를 추적하고 모니터링할 수 있는 시스템 통합
- 사용자 설정
  - 사용자가 알림 수신 설정을 조정할 수 있도록 하였다. 따라서 알림을 보내기 전 반드시 해당 설정을 확인하도록 시스템 설계 변경
- 전송률 제한
  - 사용자에게 알림을 보내는 빈도를 제한할 수 있도록 함