boostcamp
web·mobile

# Web09

나만의 길을 따라 일상을 공유하다

# 오늘의길

boostcamp
web·mobile

시연

나만의 길을 따라 일상을 공유하다

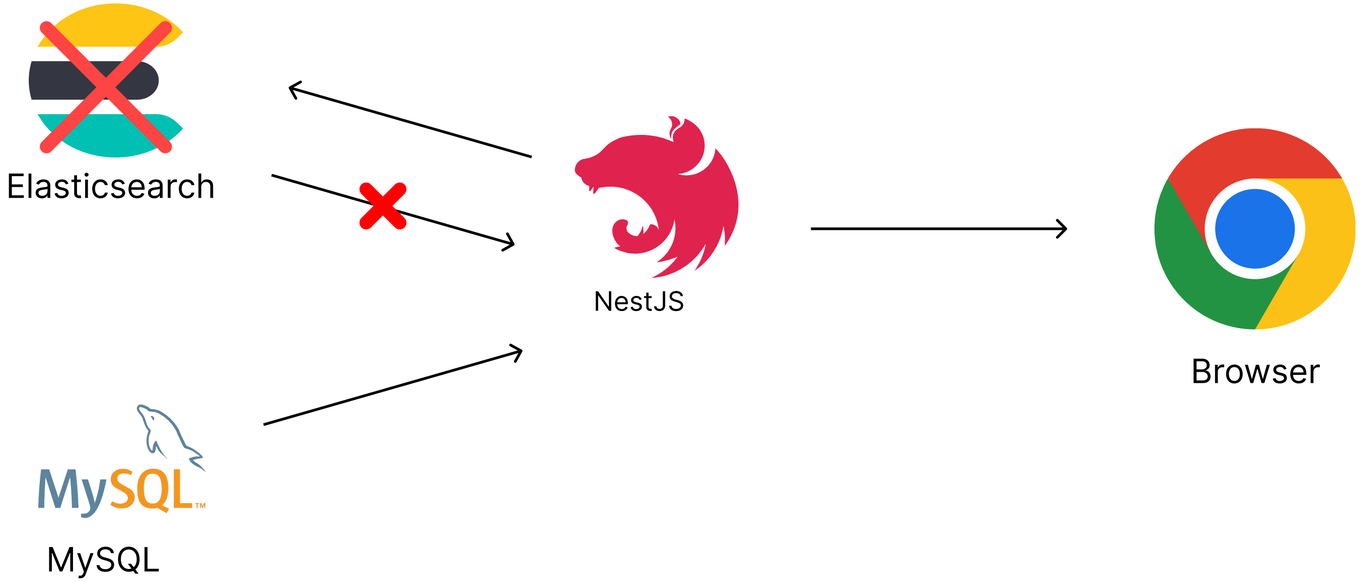
# 오늘의길

# 검색 기능 개선_WHY

| | MySQL | Elasticsearch |
|---|---|---|
| 구조 | RDBMS | NoSQL |
| 검색 방식 | 정확 매칭 (WHERE, LIKE) | 텍스트의 빈도, 형태소 등 기반 Full-Text 검색 |
| 오타 허용 | 불가능 | 가능 (Fuzziness) |
| 유사어 검색 | 불가능 | 가능 (Synonym, Tokenizer) |
| 대량 데이터 처리 | 검색 성능 저하 | 고성능 검색 |
| 복잡한 검색 쿼리 | 조건문 조합이 제한적 | 다양한 검색조건 및 가중치 조합 가능 |

# 검색 기능 개선_싱글 노드 사용에 따른 SPOF 대처

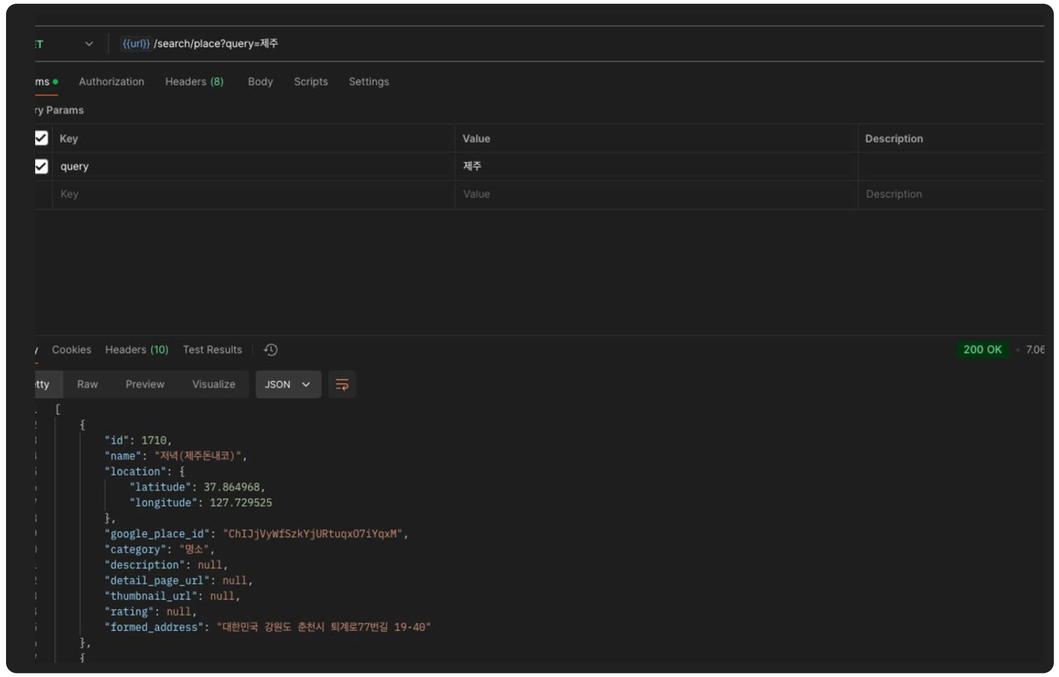
# 검색 기능 개선_싱글 노드 사용에 따른 SPOF 대처



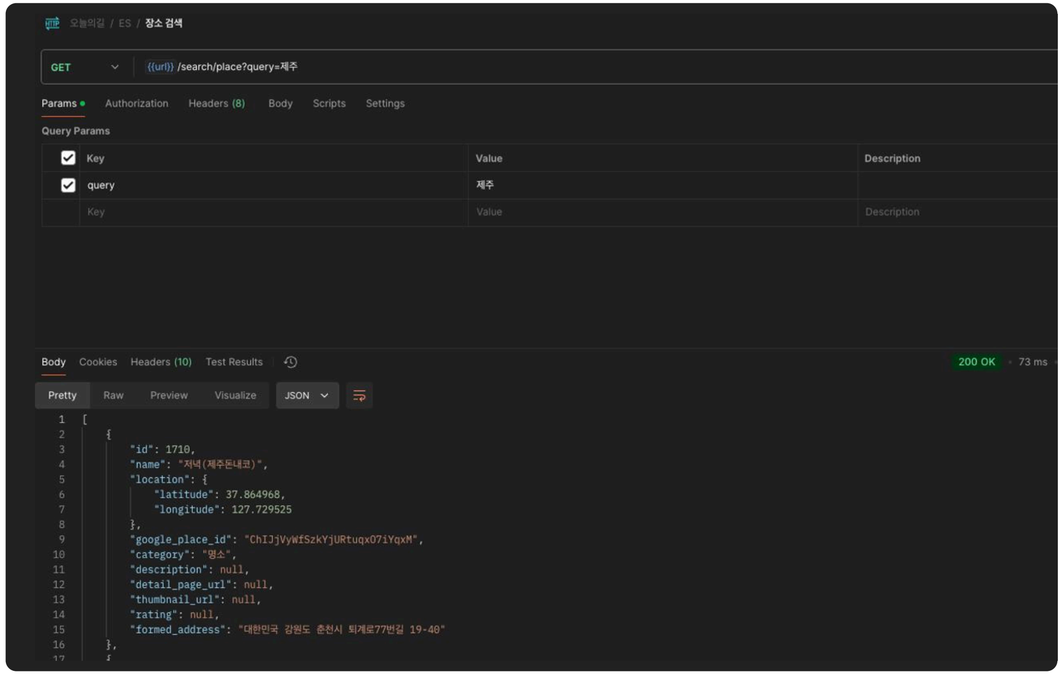
# 검색 기능 개선_싱글 노드 사용에 따른 SPOF 대처

| 이름 | 상태 | 유형 | 시작점 | 크기 | 시간 |
|---|---|---|---|---|---|
| place?query=%EB%8B%AD%EA%B0%88%EB%B9%84&page=1 | 200 | xhr | index-b0bdbd6f.js:70 | 3.4 kB | 9.07초 |
| place?query=%EB%8B%AD%EA%B0%88%EB%B9%84&page=2 | 200 | xhr | index-b0bdbd6f.js:70 | 3.0 kB | 6.71초 |
| place?query=%EB%8B%AD%EA%B0%88%EB%B9%84&page=3 | 200 | xhr | index-b0bdbd6f.js:70 | 2.8 kB | 8.84초 |
| place?query=%EB%8B%AD%EA%B0%88%EB%B9%84&page=4 | 200 | xhr | index-b0bdbd6f.js:70 | 2.5 kB | 7.53초 |

## 개선 전: 7.06s




## 개선 후: 0.073s



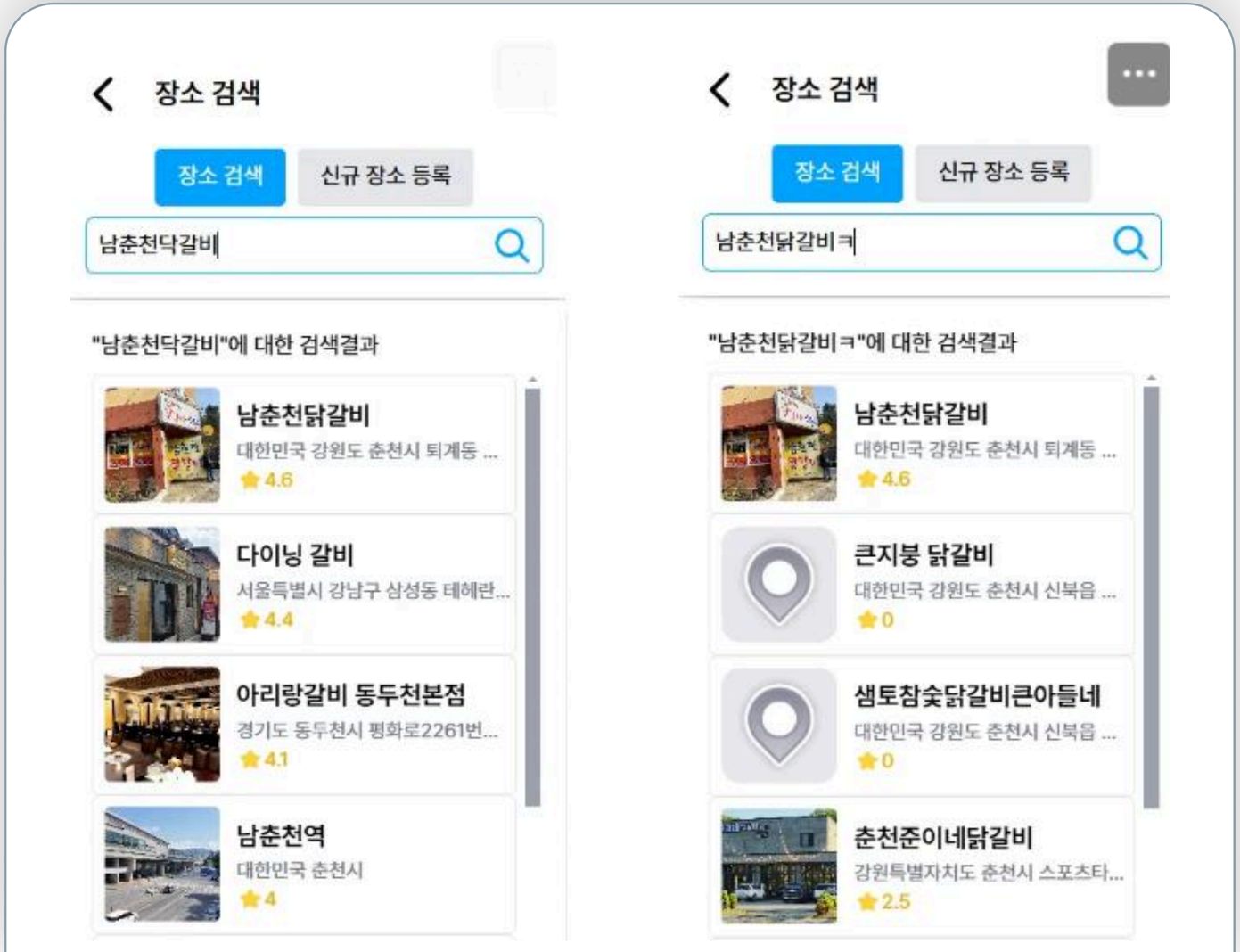
기존 : RDB를 이용한 검색

개선 : ES를 이용한 검색

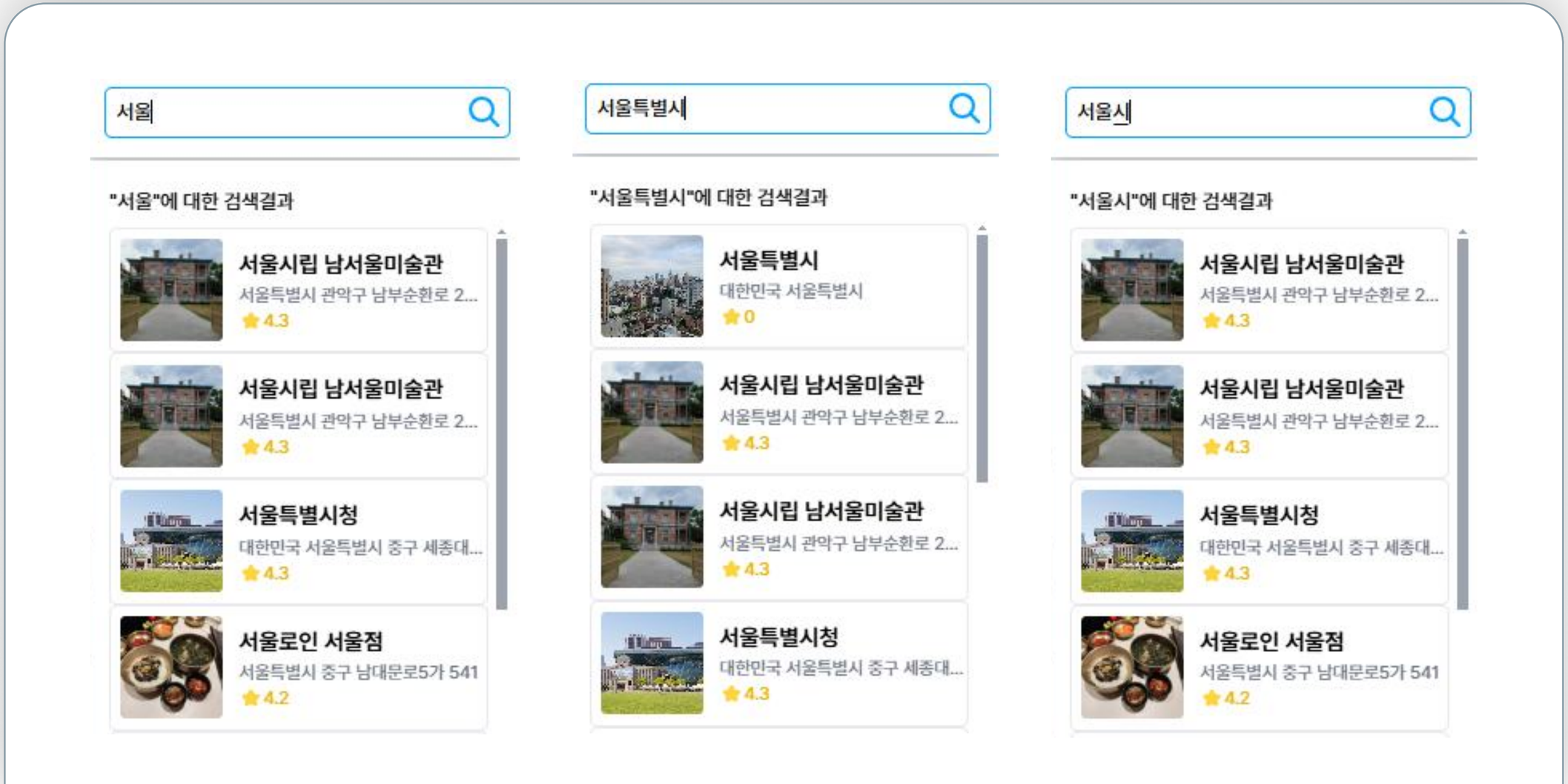
(유사 정보 추가 제공)

## 기존 : 오타 있으면 검색 결과 X

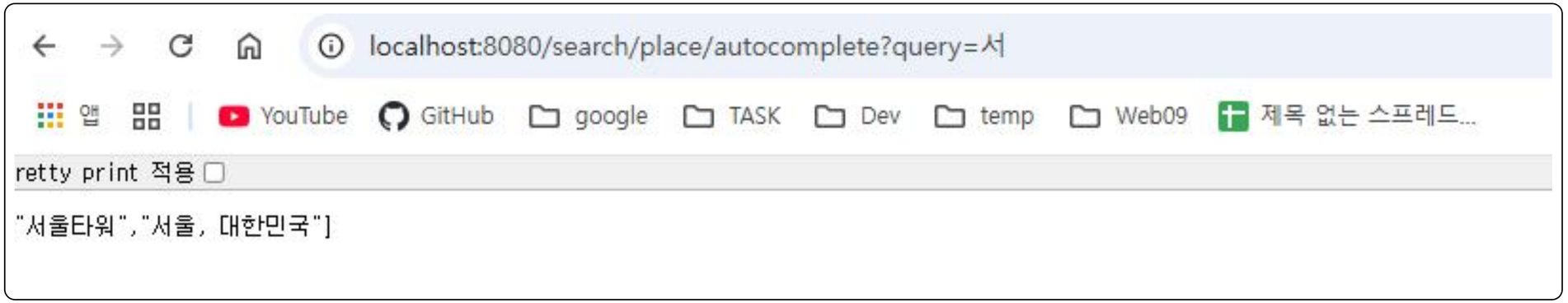
## 개선 : 오타 있어도 검색 가능

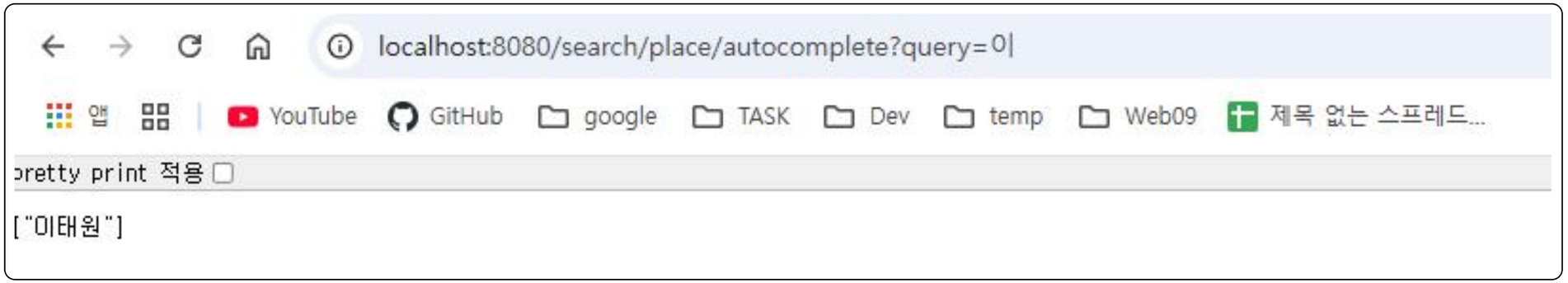


# 개선 : 지역별 동의어 처리

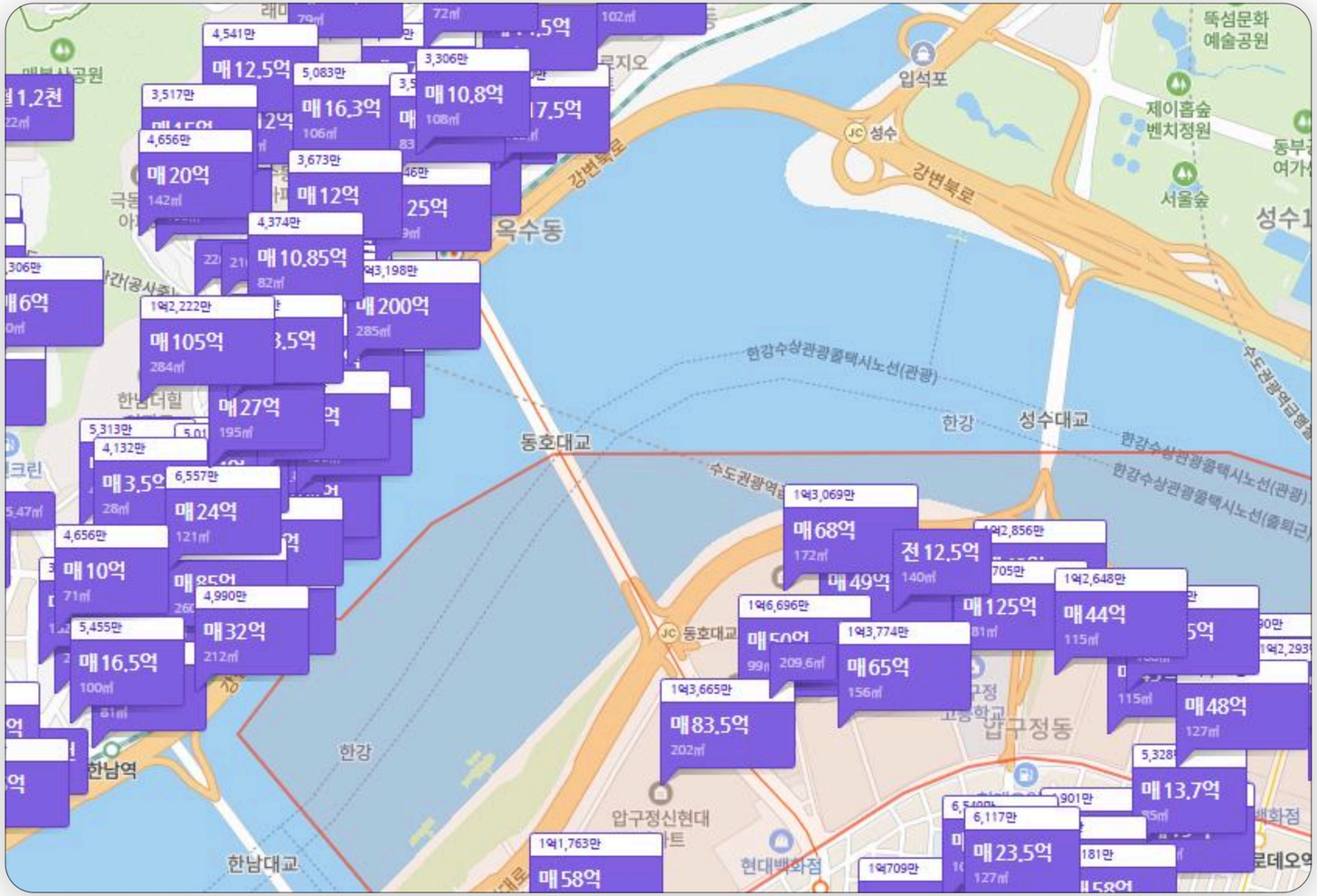

서울
"서울"에 대한 검색결과
서울시립 남서울미술관
서울특별시 관악구 남부순환로 2...
4.3
서울시립 남서울미술관
서울특별시 관악구 남부순환로 2...
4.3
서울특별시청
대한민국 서울특별시 중구 세종대...
4.3
서울로인 서울점
서울특별시 중구 남대문로5가 541
4.2
서울특별시
"서울특별시"에 대한 검색결과
서울특별시
대한민국 서울특별시
0
서울시립 남서울미술관
서울특별시 관악구 남부순환로 2...
4.3
서울시립 남서울미술관
서울특별시 관악구 남부순환로 2...
4.3
서울특별시청
대한민국 서울특별시 중구 세종대...
4.3
서울시
"서울시"에 대한 검색결과
서울시립 남서울미술관
서울특별시 관악구 남부순환로 2...
4.3
서울시립 남서울미술관
서울특별시 관악구 남부순환로 2...
4.3
서울특별시청
대한민국 서울특별시 중구 세종대...
4.3
서울로인 서울점
서울특별시 중구 남대문로5가 541
4.2

# 검색 기능 개선_자동완성


localhost:8080/search/place/autocomplete?query=서
앱
YouTube
GitHub
google
TASK
Dev
temp
Web09
제목 없는 스프레드...
retty print 적용
"서울타워","서울, 대한민국"]


localhost:8080/search/place/autocomplete?query=이
앱
YouTube
GitHub
google
TASK
Dev
temp
Web09
제목 없는 스프레드...
retty print 적용
["이태원"]

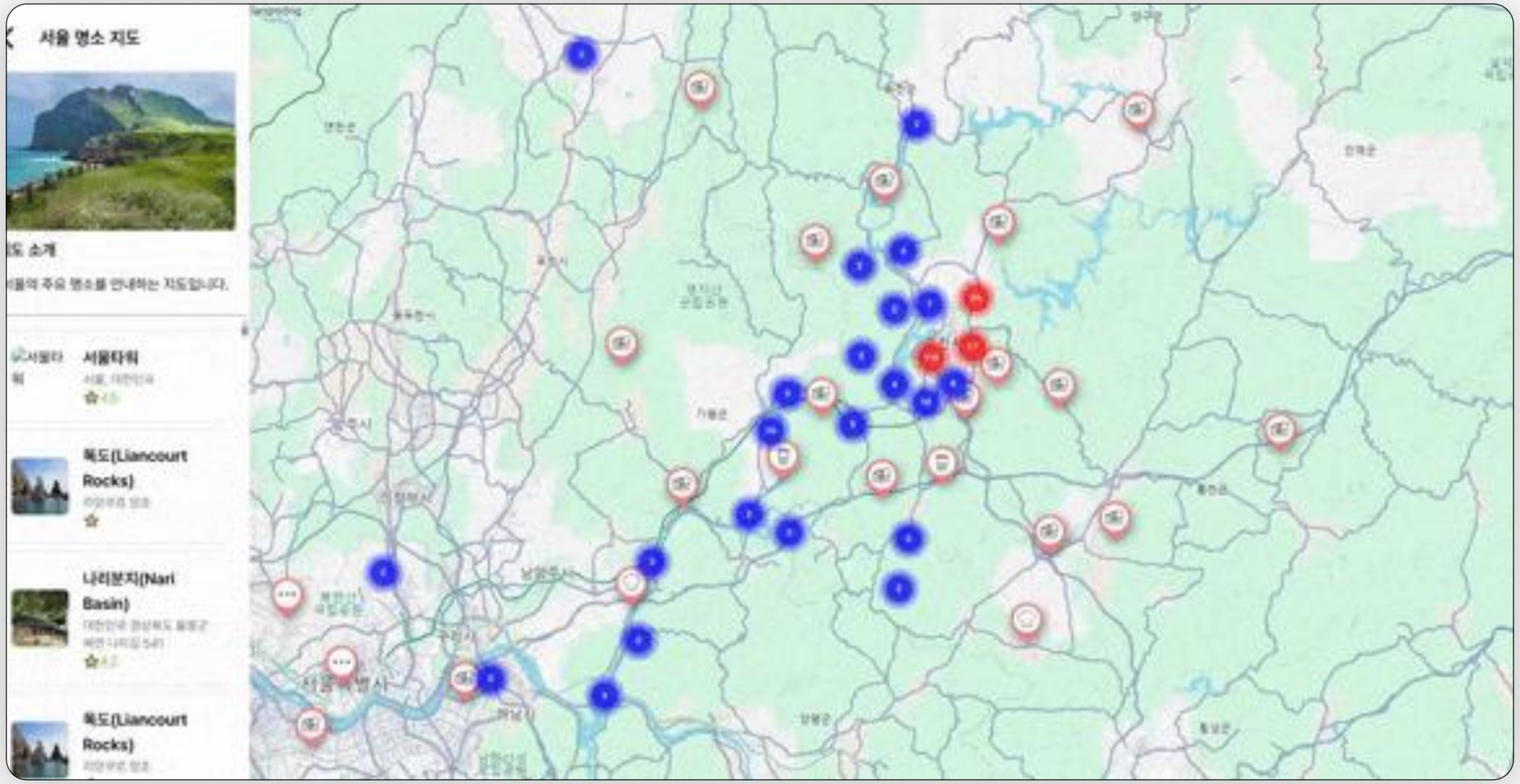
# 마커 클러스터링의 필요성



# 마커 클러스터링의 예시
# ( 네이버페이 부동산 )

# js-markerclusterer - 발생한 문제

- 매 이벤트마다 발생하는 마커 깜빡거림
- 잦은 클러스터 마커 렌더링



# js-markerclusterer - 발생한 문제

## Clusters blinking on updating its markers #864

Open aneela1191 opened this issue on Apr 1 · 3 comments

aneela1191 commented on Apr 1

Thanks for stopping by to let us know something could be better!

**PLEASE READ**

If you have a support contract with Google, please create an issue in the support console. This will ensure a timely response.

Discover additional support services for the Google Maps Platform, including developer communities, technical guidance, and expert support at the Google Maps Platform support resources page.

If your bug or feature request is not related to this particular library, please visit the Google Maps Platform issue trackers.

Check for answers on StackOverflow with the google-maps tag.

## MarkerClusterer: unnecessary rerendering of clusters when nothing has changed #801

Open ixam1 opened this issue on Nov 17, 2023 · 1 comment · May be fixed by #802

ixam1 commented on Nov 17, 2023

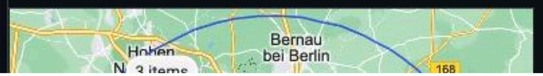
### Steps to reproduce

1. use MarkerClusterer
2. Optional: use vector maps with fractional zoom and animated cluster markers to see the effect even more
3. clusters get rerendered when zooming even when they havent changed, this causes minor flickering for default maps, but really bad experience if you use fractional zoom vector maps and animated clusters

### Video examples

**Video example with vector maps, fractional zoom and animated clusters (lots of unnecessary rerendering and animating)**

Screen.Recording.2023-11-17.at.10.24.32.mov




Assignees
No one assigned

Labels
triage me type: bug

Projects
None yet

Milestone
No milestone

Development
Successfully merging a pull re
fix: prevent unnecessary
ixam1/js-markerclusterer

**많은 사람들이 불편함을 호소하는 모습**

# 마커의 깜빡임 - 해결 방법

## 마커의 렌더링 시점 변경

markerClusterer Class 의 render 메소드 중 일부

requestAnimationFrame ⇒ setTimeOut

```
const groupMarkers: Marker[] = [];

for (const cluster of this.clusters) {
  if (cluster.marker === null) {
    continue;
  }
  if (cluster.markers?.length === 1) {
    if (!singleMarker.has(cluster.marker!)) {
      MarkerUtils.setMap(cluster.marker!, null);
    }
  } else {
    groupMarkers.push(cluster.marker!);
  }
}

this.clusters = clusters;
this.renderClusters();

// requestAnimationFrame(() => {
//   groupMarkers.forEach((marker) => {
//     MarkerUtils.setMap(marker, null);
//   });
// });

setTimeout(() => { //*
  groupMarkers.forEach((marker) => {
    MarkerUtils.setMap(marker, null);
  });
}, 25); //*
}
```

# 마커 렌더링, 잦은 DOM접근- 해결 방법

## 마커의 렌더링 조건 변경

CustomSuperClusterAlgoritm Class의
calculate 메소드

changed 변수 초기화 조건 변경

```
public calculate(input: AlgorithmInput): AlgorithmOutput {
...
    //let changed = !equal(this.state, state);
    let changed = false; //**
    if (!equal(input.markers, this.markers)) {
      this.markers = [...input.markers];

      const points = this.markers.map((marker) => {
        const position = MarkerUtils.getPosition(marker);
        const coordinates = [position.lng(), position.lat()];
        return {
          type: 'Feature' as const,
          geometry: {
            type: 'Point' as const,
            coordinates,
          },
          properties: { marker },
        };
      });
      this.superCluster.load(points);
    }

    //if (changed) {
    //  this.clusters = this.cluster(input);
    //  this.state = state;
    //}

    const newClusters = this.cluster(input); //**

    if (this.clusters.length !== newClusters.length) { //**
      this.clusters = newClusters; //**
      changed = true; //**
    } else {
      changed = false; //**
    }

    this.state = state;  //**

    return { clusters: this.clusters, changed };
  }
```

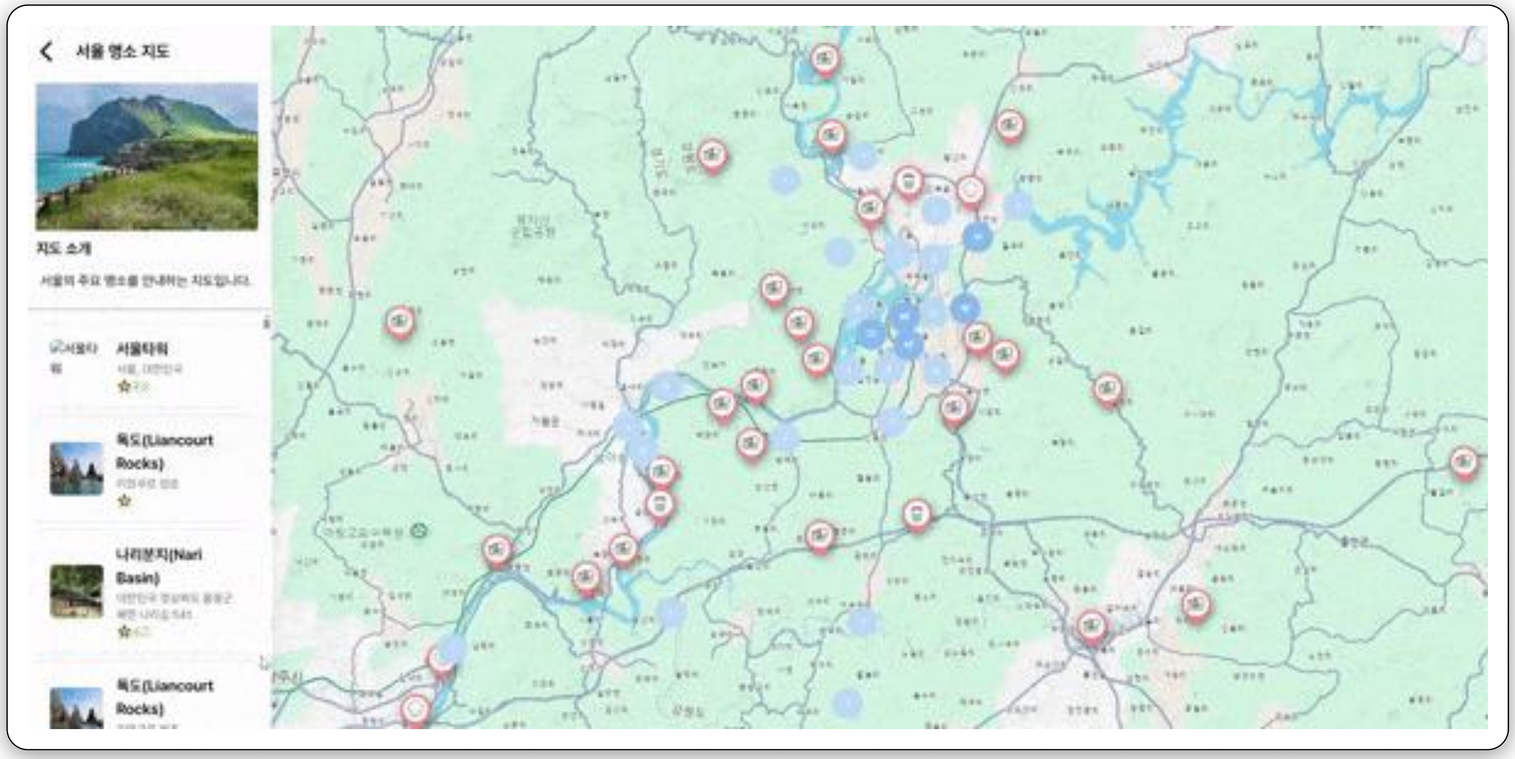
# 커스텀 마커 클러스터 성능 측정

Drag Event 측정 결과

**Origin MarkerClusterer 측정결과**
true 횟수: 11
false 횟수: 0
전체 평균 렌더링 시간: 24.68ms

**CustomMarkerClusterer 측정결과**
true 횟수: 0
false 횟수: 11
전체 평균 렌더링 시간: 1.23ms

**24.68ms ⇒ 1.23ms ( 약 20배 성능 개선 )**



# 커스텀 마커 클러스터 성능 측정

**Zoom Event 측정결과**

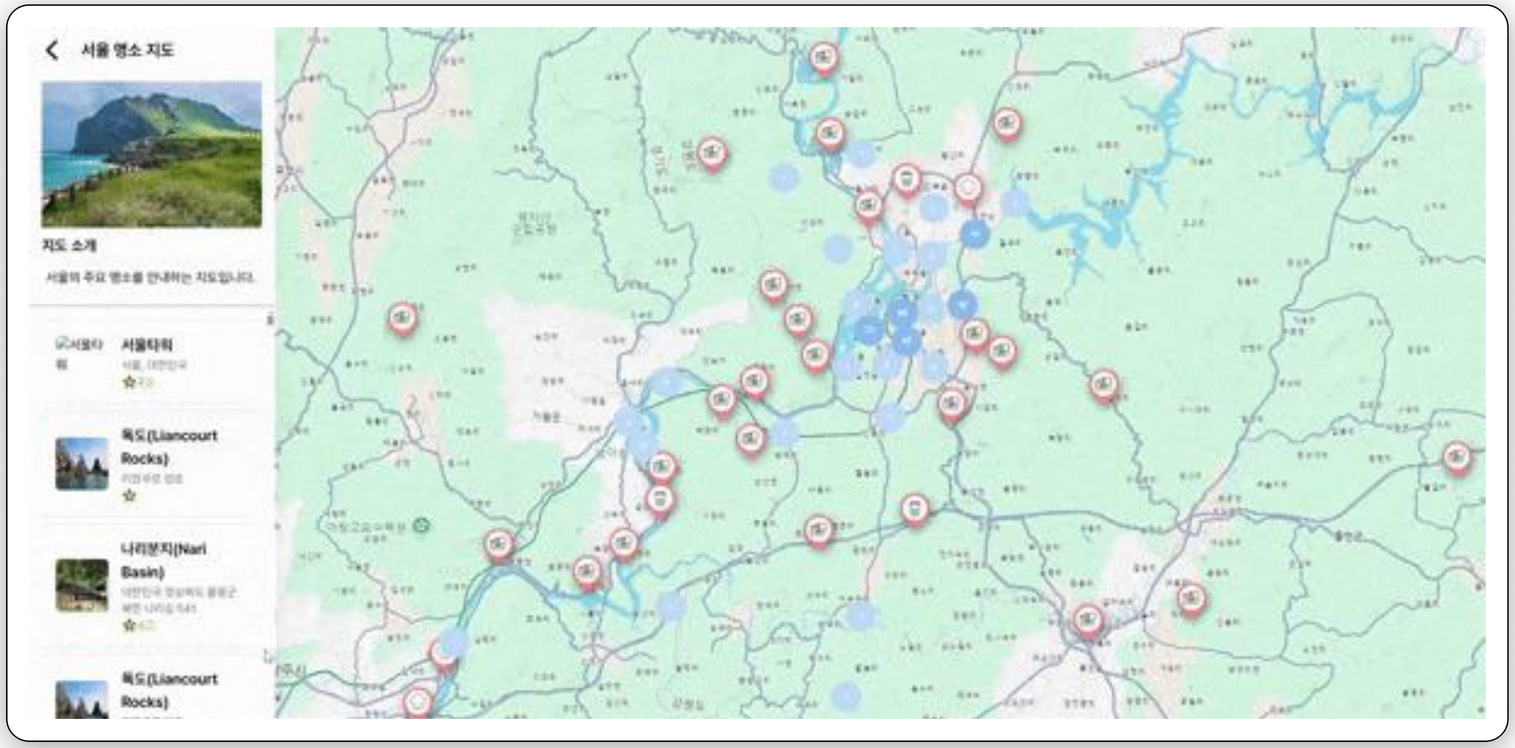
**Origin MarkerClusterer 측정결과**

true - 횟수: 43 평균 시간: 24.48ms
전체 평균 시간 : 24.48ms

**Custom MarkerClusterer 측정결과**

true - 횟수: 27 평균 시간: 28.01ms
false - 횟수: 16 평균 시간: 0.94ms
전체 평균 시간: 17.9374ms

**24.48ms ⇒ 17.937ms (약 36% 성능 개선)**

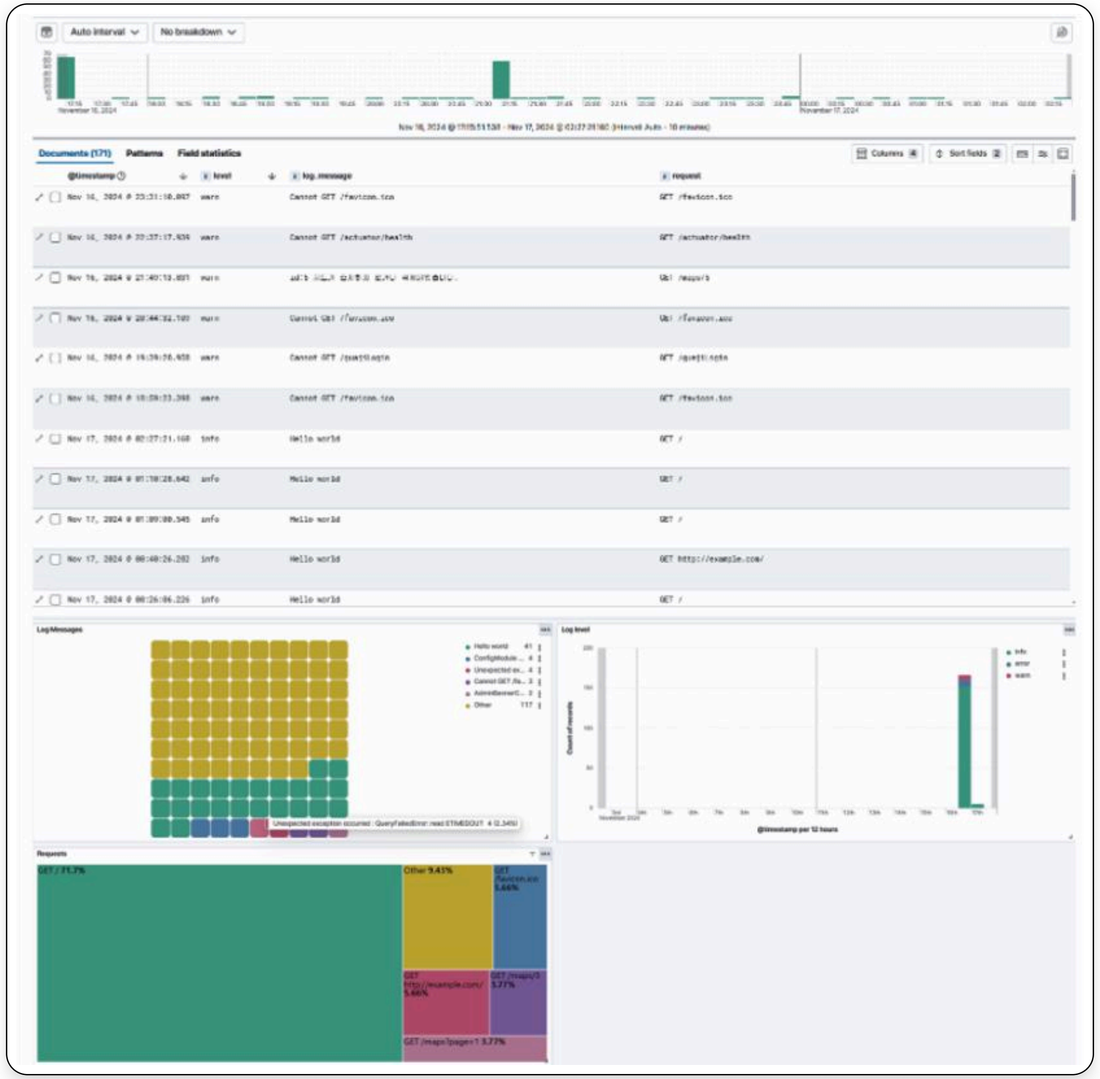


# 로깅 및 모니터링
# _WHY

```
ubuntu@daily-road:~$ docker logs nest
node:events:497
    throw er; // Unhandled 'error' event
    ^

Error: read ETIMEDOUT
    at TCP.onStreamRead (node:internal/stream_base_commons:218:20)
    at TCP.callbackTrampoline (node:internal/async_hooks:130:17)
Emitted 'error' event on Socket instance at:
    at emitErrorNT (node:internal/streams/destroy:169:8)
    at emitErrorCloseNT (node:internal/streams/destroy:128:3)
    at process.processTicksAndRejections (node:internal/process/task_queues:82:21) {
  errno: -110,
  code: 'ETIMEDOUT',
  syscall: 'read'
}

Node.js v20.18.0
```



# 로깅 및 모니터링_PinoLogger

Miensoap 3 weeks ago Maintainer

## 로깅 라이브러리 `pino` 를 제안합니다!

winston vs pino

**장점**

1. 고성능/저비용 로깅
2. 구조화된 json 로그 제공

**단점**

1. 출력 옵션 제한 (파일 저장)
2. 포맷 커스터마이징 제한 ( json 사용 )

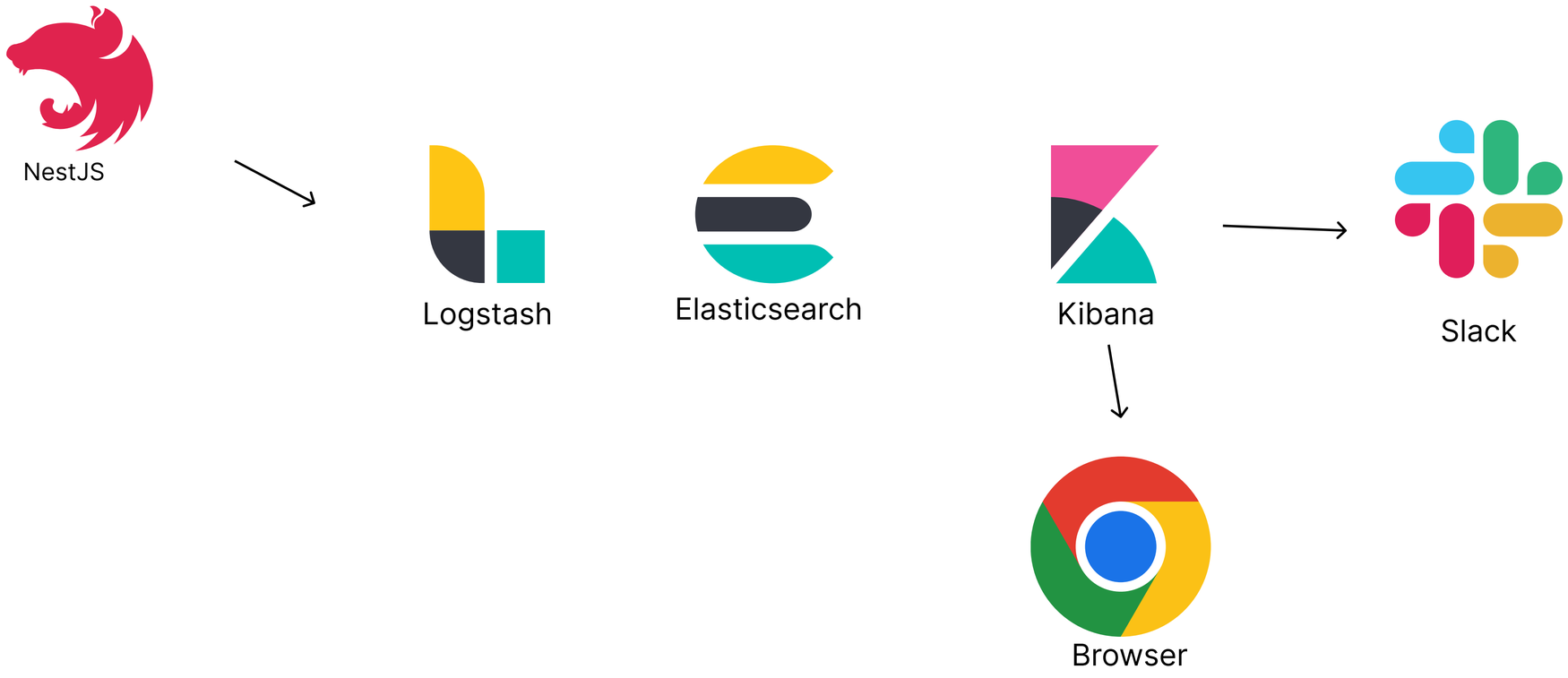
단점인 출력 옵션 제한은 ELK 연동으로 LogStash 로 수집해 해결
개발 환경에서 `pino-pretty` 플러그인 사용으로 쉽게 읽을 수 있음

### Development Formatting

The `pino-pretty` module can be used to format logs during development:

```
> node examples/basic.js | pino-pretty

[08:41:11.655] INFO (57728): hello world
[08:41:11.656] ERROR (57728): this is at error level
[08:41:11.656] INFO (57728): the answer is 42
[08:41:11.656] INFO (57728): hello world
    obj: 42
[08:41:11.656] INFO (57728): hello world
    obj: 42
    b: 2
[08:41:11.656] INFO (57728): nested
    nested: {
      "obj": 42
    }
[08:41:11.656] ERROR (57728): an error
    err: {
      "type": "Error",
      "message": "an error",
      "stack":
          Error: an error
              at Object.<anonymous> (C:\Users\frazer.smith\Documents\Repositories\forks\pino\examples\basic.js:21:12)
              at Module._compile (node:internal/modules/cjs/loader:1105:14)
```


NestJS
Logstash
Elasticsearch
Kibana
Slack
Browser

# 로깅 및 모니터링_결과

오후 6:39 **Log Management Alert** 앱 🚨 긴급 로그 발생

⭐ 로그 레벨: `warn`

📄 내용:

```
Cannot GET
/asdf%E3%85%81%E3%84%B9%E3%85%87%E3%84%B4%E3%85%81%E3%84%B9%E3%85%87%
%87%E3%85%81%E3%84%B9%E3%85%87%E3%84%B4%E3%85%81
```

⏰ 발생 시간: `2024년 12월 04일 18:39:37`

오후 6:57 **오길봇** 앱

**배포 결과**

**배포 상태:** 배포 성공
**배포 시간:** 2024-12-04 18:57:31
**배포 전 로그:** 배포 전 상세 로그 보기
**배포 로그:** 배포 상세 로그 보기

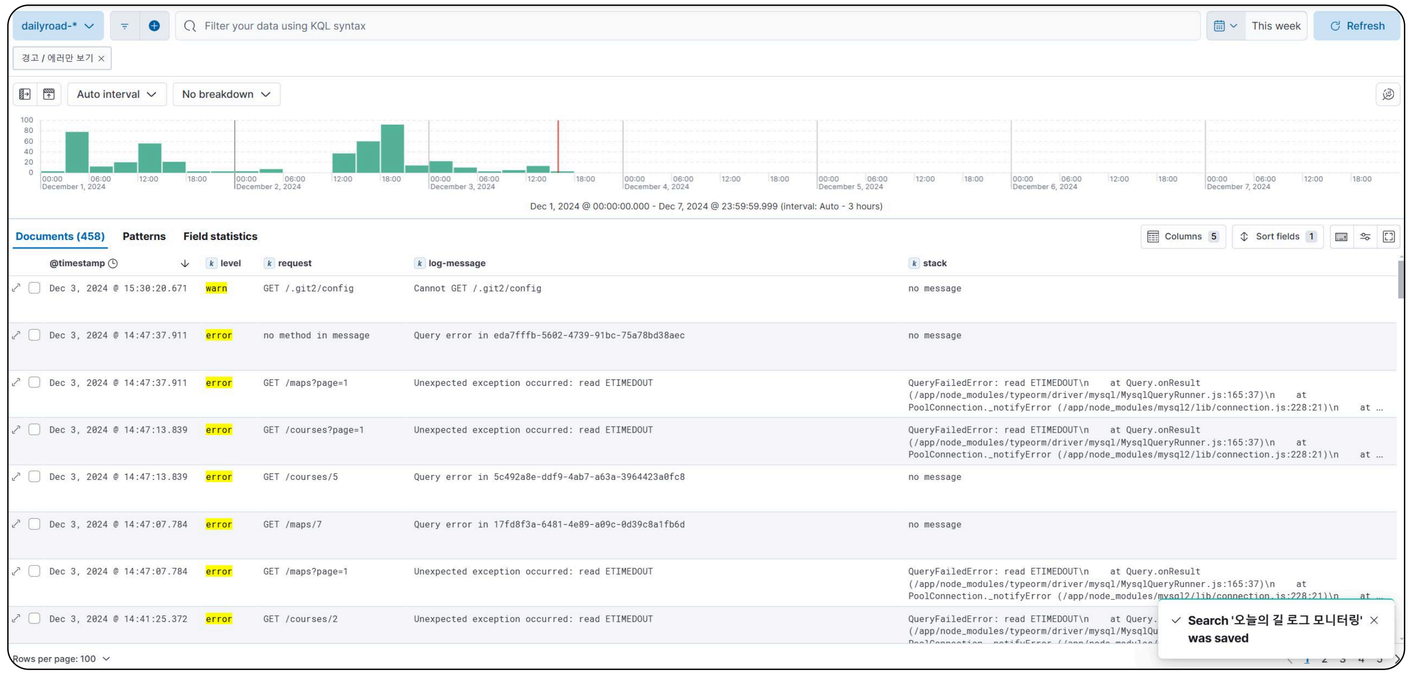
오후 6:58 **Log Management Alert** 앱 🚨 긴급 로그 발생

⭐ 로그 레벨: `error`

📄 내용:

```
AppController.getHello의 에러 로그입니다.
```

⏰ 발생 시간: `2024년 12월 04일 18:58:27`



## 테스트 코드 왜 작성해야할까?

테스트 코드는 손으로, 눈으로 기능 검증하는 것 보다

더 빠르고 자주 수행할 수 있기 때문에

제품을 자주 배포할 수 있게 해준다

# 테스트_ 레이어별 테스트 대상

## Service 의 역할은 무엇인가?

1. 핵심 로직 처리
   a. 데이터 조작 및 연산
   b. 외부 API 통신 및 통합 ( `Injectable` 로 구분 가능)
2. 트랜잭션 관리
   a. ex) 계좌이체 로직에서 내가 출금했는데 상대방이 입금되지 않았다면?
3. 예외 처리
   a. `if (~~) throw new Error()`
4. 응답 DTO 생성

## Repository의 역할은 무엇인가?

1. DB와 상호작용
   a. `ORM` 의 메소드 혹은 `SQL` 쿼리 통해 알맞은 데이터를 가져오는지
   b. `ORM` 의 메소드를 이용하여 만든 커스텀 메소드가 잘 작동하는지

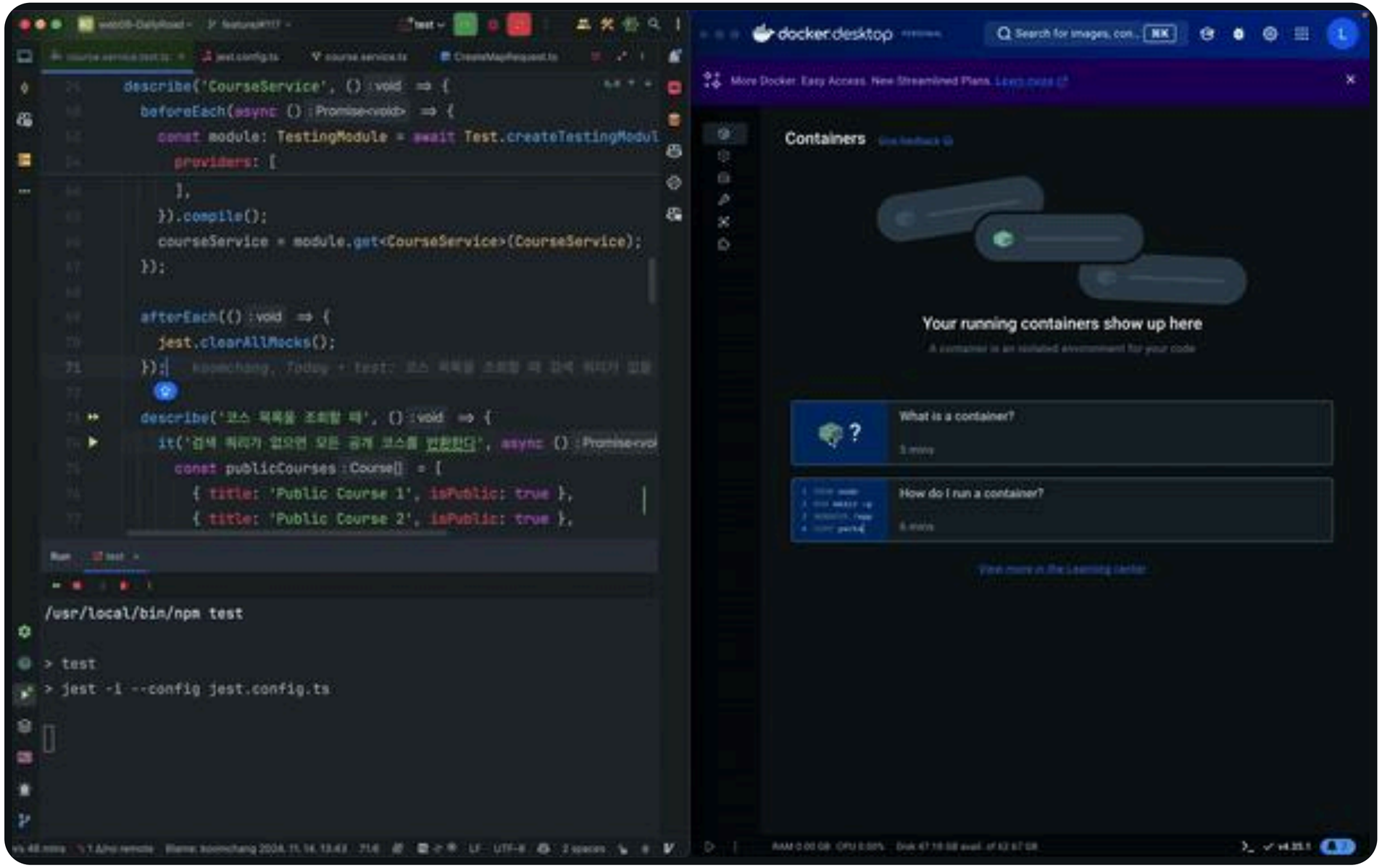
> DB와 실제로 커넥션하며 테스트 해야겠다!
>
> CI 과정에서 테스트까지 고려한다면 `Docker` 로 간단하게 DB 컨테이너를 만들어 테스트 환경을 구성해주는 것도 괜찮을 것 같다.

## Controller 의 역할은 무엇인가?

1. 요청 데이터의 유효성 검사
   a. 필수 파라미터가 누락되지는 않았는지 확인
2. 미들웨어와 인터셉터
   a. 인증, 로깅 등 확인
3. Parser
   a. `HTTP Request` 의 `Start line` , `Headers` , `Body` 를 파싱
4. 쿠키
   a. 쿠키 데이터를 파싱하고 필요한 경우 응답 쿠키에 설정을 추가
5. Serialize
   a. 파싱한 데이터를 `DTO` 객체로 맵핑
   b. DTO 검증 (DTO 클래스에서 `class-validator` 를 이용하거나 검증 로직 작성)
6. Deserialize
   a. `DTO` 객체를 `Content-Type` 에 맞게  응답 객체 생성
7. 성공 응답
   a. 원하는 응답 내용인지 확인
      i. `Header`
      ii. `StatusCode`
      iii. `Body`
8. 예외 및 에러 응답
   a. 원하는 예외 응답인지 확인
      i. `Header`
      ii. `StatusCode`
      iii. `Body`

# 테스트

## _ 컨테이너 사용




### 컨테이너를 도입하며 생긴 문제 상황

1. 컨테이너가 중복 생성 되어 과도한 리소스 점유
2. 테스트 병렬 실행으로 동시성 문제 발생

# 테스트_ 백엔드 테스트 규칙

| | 런던파 (Mockist) | 고전파 (Classicist) |
|---|---|---|
| **테스트 방법** | 테스트 대상 객체를 중심으로<br>Mock을 사용하여 의존성을 대체하여 테스트 | 테스트 대상 객체와 모든 실제 의존성을 함께 테스트 |
| **장점** | • 테스트 대상 객체만의 문제를 빠르게 식별 가능<br>• 테스트 실행 속도가 빠름 | • 실제 코드와 유사한 방식으로 작동해 신뢰도가 높음<br>• 요소들이 상호작용할 때 잘 작동하는지 확인 가능 |
| **단점** | • 실제 시스템과의 차이가 있음 | • 테스트 속도가 느림 |

팀원이 작성한 코드를 다른 팀원이 테스트 → 서로의 코드를 검토 및 이해

# 테스트 _ 결과

```
Test Suites: 9 passed, 9 total
Tests:       117 passed, 117 total
```

## All files

**74.97%** Statements `1366/1822` **32.64%** Branches `79/242` **55.97%** Functions `225/402` **74.03%** Lines `1229/1660`

# Q & A

어떤 내용을 로깅?

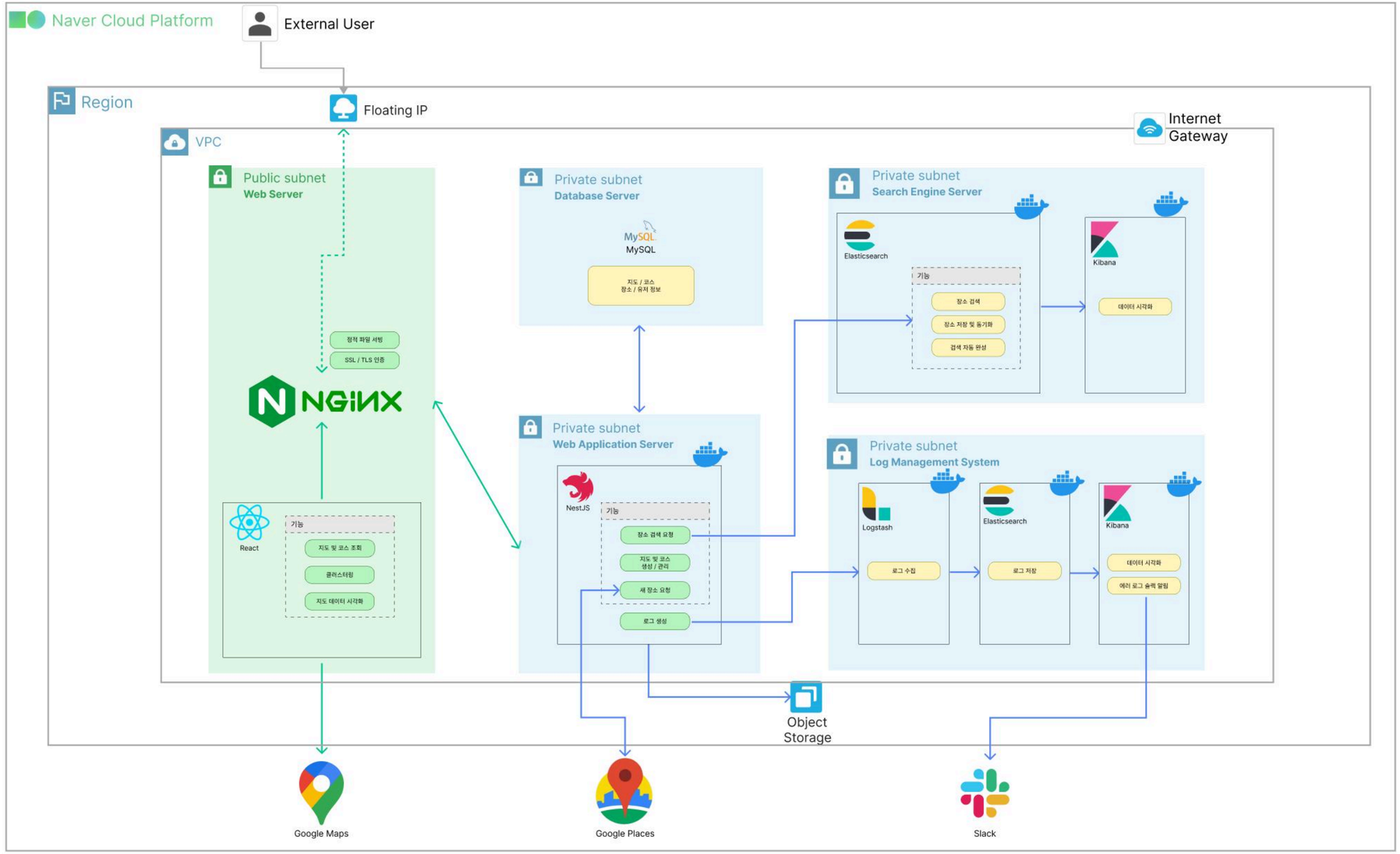
로그 레벨은?

클러스터링 구현 방법?

배포 알림 도입 이유?

# 부록

## _ 서비스 아키텍처


Naver Cloud Platform
External User
Region
Floating IP
Internet Gateway
VPC
Public subnet
Web Server
정적 파일 서빙
SSL / TLS 인증
NGINX
React
기능
지도 및 코스 조회
클러스터링
지도 데이터 시각화
Private subnet
Database Server
MySQL
MySQL
지도 / 코스
장소 / 유저 정보
Private subnet
Web Application Server
NestJS
기능
장소 검색 요청
지도 및 코스
생성 / 관리
새 장소 요청
로그 생성
Private subnet
Search Engine Server
Elasticsearch
기능
장소 검색
장소 저장 및 동기화
검색 자동 완성
Kibana
데이터 시각화
Private subnet
Log Management System
Logstash
로그 수집
Elasticsearch
로그 저장
Kibana
데이터 시각화
에러 로그 슬랙 알림
Object Storage
Google Maps
Google Places
Slack

# 부록

## _ 로깅에서 키바나 필터 적용

Documents (36) Patterns Field statistics

| @timestamp | log_level | log_msg | request | @timestamp_seoul |
|---|---|---|---|---|
| Dec 4, 2024 @ 21:13:54.354 | warn | 만료된 토큰입니다. | GET /users/info | 2024년 12월 04일 21:13:54 |
| Dec 4, 2024 @ 21:12:25.205 | warn | Cannot GET /geoserver/web/ | GET /geoserver/web/ | 2024년 12월 04일 21:12:25 |
| Dec 4, 2024 @ 21:11:44.062 | warn | Cannot GET /favicon.ico | GET /favicon.ico | 2024년 12월 04일 21:11:44 |
| Dec 4, 2024 @ 21:00:55.269 | warn | 리프레시 토큰이 없습니다. | POST /oauth/refresh | 2024년 12월 04일 21:00:55 |
| Dec 4, 2024 @ 21:00:55.268 | warn | 만료된 토큰입니다. | GET /users/info | 2024년 12월 04일 21:00:55 |
| Dec 4, 2024 @ 20:52:41.684 | warn | [21] 지도가 존재하지 않거나 삭제되었습니다. | GET /maps/21 | 2024년 12월 04일 20:52:41 |
| Dec 4, 2024 @ 20:52:40.662 | warn | [21] 지도가 존재하지 않거나 삭제되었습니다. | GET /maps/21 | 2024년 12월 04일 20:52:40 |
| Dec 4, 2024 @ 20:11:18.922 | warn | 동일한 장소는 연속된 순서로 추가할 수 없습니다. | PUT /courses/6/places | 2024년 12월 04일 20:11:18 |

# 로깅 및 모니터링_로깅 레벨 결정 기준

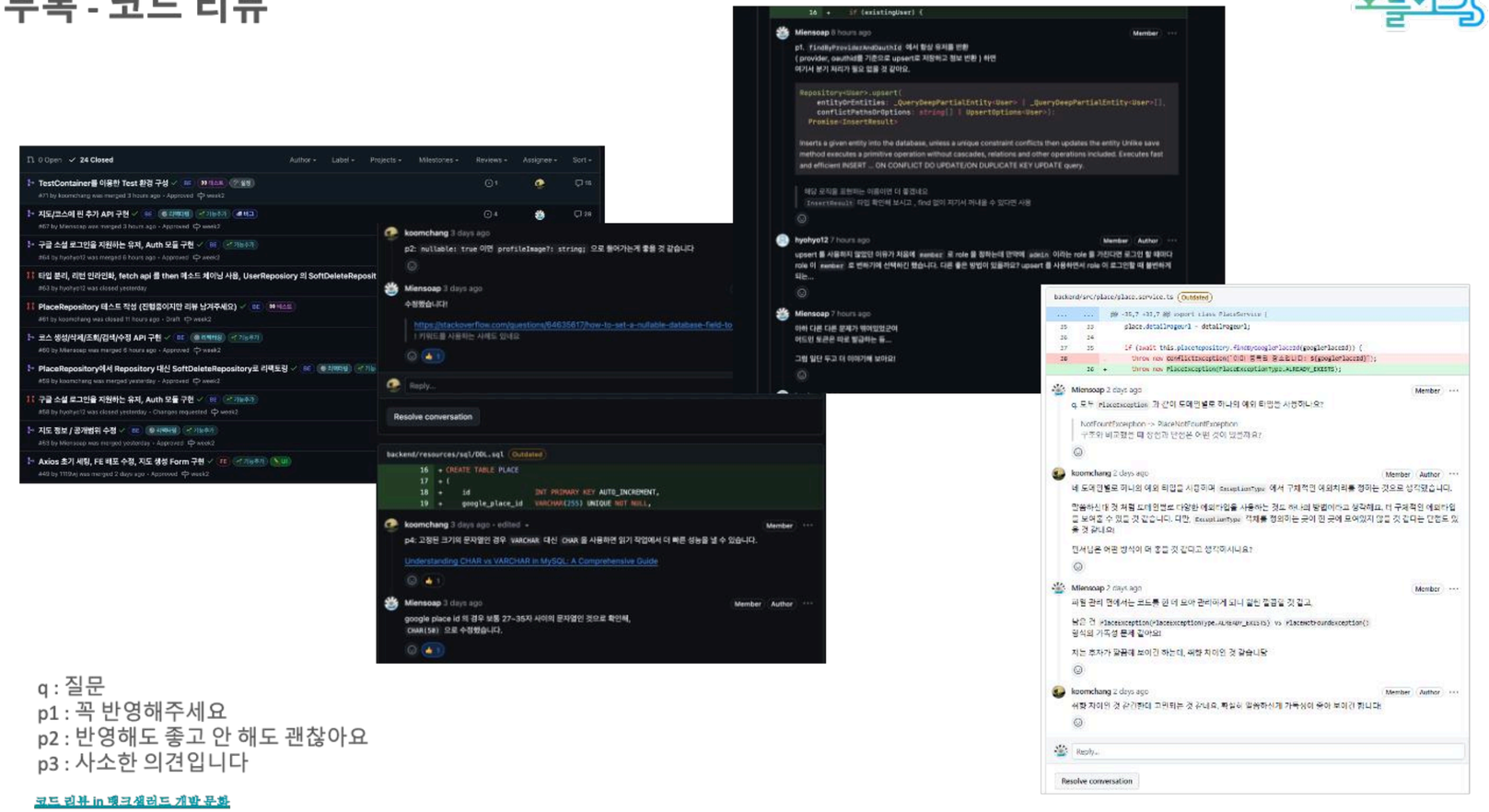
WARN : 400번대 사용자 에러

ERROR : 500 번대 서버 에러 -> 즉시 조치

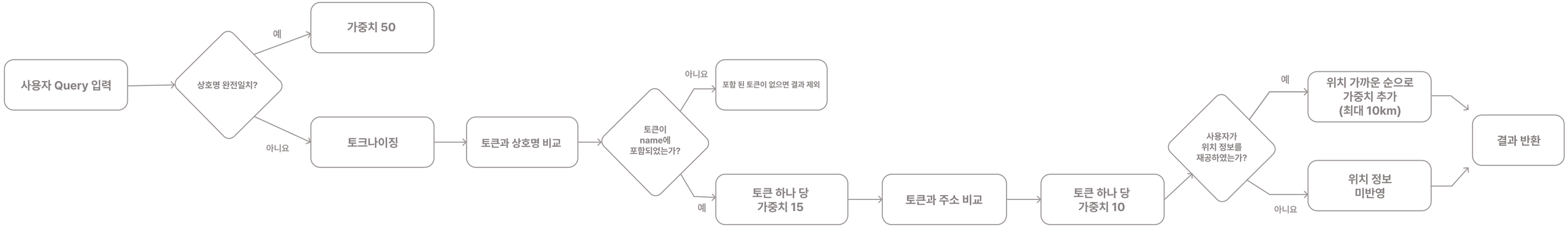
# 부록

## _ 코드 리뷰

### 부록 - 코드 리뷰




q : 질문
p1 : 꼭 반영해주세요
p2 : 반영해도 좋고 안 해도 괜찮아요
p3 : 사소한 의견입니다

코드 리뷰 in 백크엔드 개발 문화

# 검색 기능 개선 - 검색 로직 설계 (가중치 적용)