

# toPangyo

## 동세권

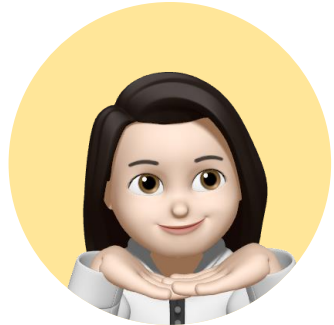
# 팀원



우명규

---

Team Leader  
&  
Front-end



류나연

---

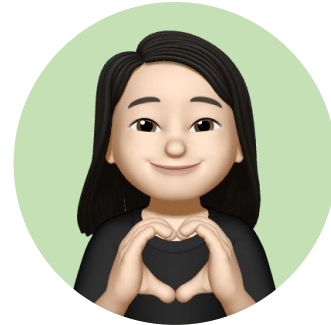
Front-end



선원종

---

Back-end



전혜지

---

Back-end



최성민

---

Back-end

# 최악의 팀

“이런 PMP문서를 가져온 건 이 팀 밖에 없는데?  
최악이에요.”



# 고난

: 괴로움과 어려움을 아울러 이르는 말.

# 고난 1 : 글로벌한 우리 팀

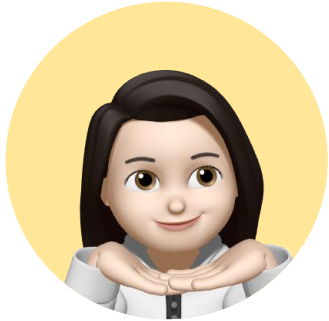
한국



우명규

Team Leader  
&  
Front-end

뉴질랜드



류나연

Front-end

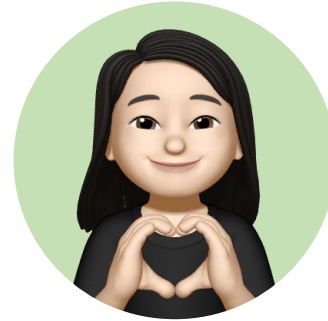
뉴질랜드



선원종

Back-end

뉴질랜드



전혜지

Back-end

미국



최성민

Back-end

## 고난 2 : 이거 또 뭔데 ?




redis

NGINX

**Service worker**

**[ 잘 ] 하고 싶다.**

toPangyo

동세권 

사용자들이 여러 장소를 함께 가도록 도와주는  
플레이스 매칭 플랫폼



# 시연영상

**성공적인 프로젝트가 되기 위한..**

**해결방법**

# 방법 1 : 목표의 재설정 (이전)

동시  
세권

## 02 팀의 목표

• MSA 아키텍처를 통해 각각의 서비스를 독립적으로 연결하여 서비스 확장성을 고려하고 장애에 대하여 코드 전체에 영향을 미치는 문제가 없도록 설계하기 ( 실제로 상용화할때 유저들의 트래픽을 처리하는데 있어서 큰 장애를 미리 방지하기 위해 )

• 한 단위의 작업이 끝나면 자신의 브랜치에 작업내용 올리기, 올릴 시 커밋 양식을 통일하여 커밋 메세지만을 보고 어떤 내용이 수정되고 추가되었는지 모든 팀원이 파악가능하도록 작성하기

• 같이 일하고싶은 개발자 되기

### 최성민

#### 1.RESTful api 중심의 비즈니스 서버를 구성한다.

- api설계시 고려해야하는 컨텍스트, 네트워크에 효율적인 웹 api 설계법에 대해 서치한다.
- 이를 바탕으로 "동세권"의 서버를 설계한다.
- 이러한 설계가 적절한지에 대해 검토해본 후 멘토분들의 피드백을 적극적으로 활용한다.

#### 2.배포 과정 파악과 아키텍처 설계, 구현하기

- 프로젝트에 대한 웹 서버 배포 과정을 파악한다.
- ( Nginx, 도커, aws등에 대한 역할과 이론을 설명할 수 있는 상태까지 도달한다. )
- 내용에 대해 팀원들과 공유하며 이를 바탕으로 아키텍처를 설계한다.
- 설계된 아키텍처가 합당한지에 대한 피드백을 진행한다.
- 검색과 멘토분들을 적극적으로 활용하여 해당 설계에 대해 피드백받는다.
- 아키텍처의 설계는 MSA아키텍처를 지향한다.
- ( 이때 서버를 나누는 단위, 서버간 통신 등에 대해 자세히 파악한다. )

#### 3.철저한 계획에 기반한 스키마 모델 구축

- 현재 익숙하지 않은 RDBMS와 비교적 익숙한 NoSQL DB를 고루 생각하여 목적에 맞는 데이터베이스를 구축한다.

#### rule :

- 각 서치의 단계는 하루를 넘기지 않는다.
- 서치한 것에 대해서는 꼭 개인 개발 과정에 기록하며, 기록은 flow차트 형식을 지향한다.
- 설계 및 기능 구현등에 대해 정리가 완료된 항목에 대해서는 즉각적으로 팀원들과 공유한다.

### 우명규

#### 1.MVC패턴 아키텍처 파악하기

- 리액트에서 사용되는 MVC 패턴의 아키텍처를 파악하여 페이지 설계하기
- MVC의 단점으로 인해 생기는 Side Effect를 방지할 수 있는 방법 터득하기

#### 2.백엔드와의 협업방법 flow 파악하기

- 협업하는 방법을 터득하고 API, git 커밋메세지, To do list 등 PM에 필수적인 문서들 정의하는 법 알기

#### 3.필요한 라이브러리 사용법 익히기

- 지도 API를 가져와 DB에 저장되어있는 데이터를 가공하여 처리하기
- 상태관리 라이브러리 Redux로 state를 관리하고 해당 컴포넌트에 필요한 데이터 추출하기

### 전혜지

#### 1.DB 쿼리 효율적이고 직관적으로 짤 수 있는 능력 갖추기.

- 가독성과 성능을 위해 WHERE 절 FROM 절 순서 확실히 파악하여 작성하기.
- 불필요한 조건 빼기, 검색 조건 간단히 작성하기.

#### 2.git을 통해 으로 프로그램 관리하는 능력을 갖추기.

- 명령어 리스트 정리해 놓고 쓰면서 자연스럽게 외우기.
- commit 메세지만으로 기능 확실하게 알 수 있도록 쓰기.

#### 3.문서 작성하기

- 동일한 오류가 다시 났을 때, 무의미한 검색 시간 줄이기 위해 오류에 관련한 문서 작성하기.
- 전체적인 개발 진도 파악 위해 진행 정도, 상황 기록하기.

### 선원중

#### 일회용 프로젝트가 아닌 지속적 기록과 문서화로 회고할 수 있는 나만의 지속가능한 자산을 만들자

\_\_실제 서비스 되는 것까지 고려한 정교화된 설계 필요, 지속적관리

#### 세부사항)

- 필드 명세설정, 관계설정을 생각하며 모델링을 작성하고 정규화 와 그 과정을 기록
- 매일 모닝 스크립 회의 시 팀원들이 이해할 수 있을 정도의 설명 준비
- MSA를 모든 파트가 아닌 중요도가 높은 독립된 서비스가 필요한 곳에 적용해보기 (채팅 서버)
- 팀원들과 작은것 부터 자주 소통하고 항상 긍정적인 모습 보여주기
- 목표를 최대한 유지하지만 세 시간에 완성될 수 있게 구현하는것 (유연한 사고, 계획)

#### 추가사항)

- 목표달성 위한 발전 - 지속적으로 기록하기 및 각 단계별로 회고 진행
- 최선을 다해 구현하고 리팩토링 진행, 부족한 부분은 지도를 요청해보기
- 적절한 검색법에 대한 이해와 좋은 자료가 있으며 정리해서 내 것으로 만들기 및 팀원 공유

### 류나연

#### 1.프로그램 설계 조직화 및 클린 코드 작성

- 재사용할 컴포넌트와 아닌 컴포넌트를 구별하며 프로그램을 설계 조직하기
- 구상도 작성 및 피드백하는 단계를 통해 이를 조직화하여 구분하기
- 코드 작성 후 최소 2회 확인하며 코드 내 오류, 실수 확인 및 어려웠던 점 회고

#### 2.Redux, Scss, jwt 개념 배우고 정리 및 적용해보기

- 각 기능 또는 개념 별 최소 3개의 영상 또는 문서를 정독하기
- 그 후 이해 대한 정리 문서 또는 실습 파일을 만들기
- 해당 과정을 통해 선이해한뒤 이를 프로젝트에 적용하기

#### 3.기록 습관 만들기

- 매일 to do list를 작성하며 기록을 몸에 익숙하게 만들기
- 해당 리스트를 바탕으로 일정을 진행하고 피드백하며 체계적인 습관 만들기
- 또한 커밋 양식에 맞춰 작성하며 협업을 위한 기록을 습관화하기

# 방법 1 : 목표의 재설정 (이후)

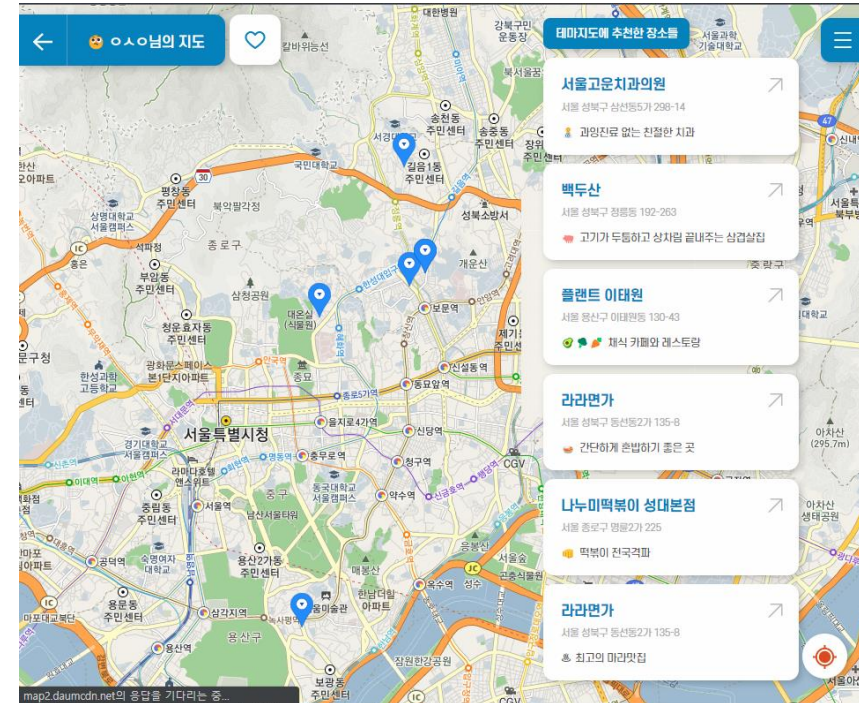
## 클라이언트

: 지도 api를 활용한 가독성 있는 UI 화면 구성

## 서버

: 경험해보지 못한 소켓 통신을 활용한 메신저 프로젝트 (ex디스코드)

자료조사 ?



## 진짜서울

: 지도 위에서 원하는 것에 부합하는 장소를 보여주는 큐레이팅 사이트.

# 방법 1 : 목표의 재설정 (이후)


진짜 서울

+

서버의 니즈

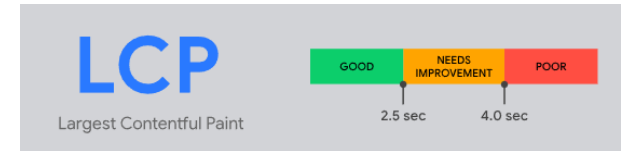


니즈를 충족하는 플랫폼 기획

동세권 

## 도출된 팀의 목표

우수한 사용자 경험을 제공하기 위해서는  
2.5s의 LCP를 가져야함



LCP  
2.5초 이내로  
최적화

# 방법 1 : 목표의 재설정 (이후)

## 공통: 협업 경험



우명규

1. React의 Virtual DOM만을 활용해 프로젝트를 개발할 것이고 컴포넌트가 여러 번 렌더링되지 않도록 useEffect의 clean-up함수로 초기화 값을 지정하여 메모리 누수 방지하기
2. API를 사용하는 코드는 async & await으로 비동기 처리를 하여 응답 대기시간을 줄이고 문제가 있는 요청이 들어와도 try-catch문으로 사용자에게 정상적인 페이지 응답하기
3. state는 Redux로 관리하여 과도한 props drilling을 방지하고 각 state의 Reducer는 파일을 따로 관리하여 store에 적용시키기



류나연

1. 카카오 지도 api를 바탕으로 geolocation api를 사용하여 현재위치를 받아오며 해당 위치를 중심좌표로 지도의 확대 레벨을 구하고 함수를 사용하여 인포윈도우에 해당하는 마커에 대해 생성과 표기를 하고 접속위치를 변경하는 로직 구현하기
2. 매일 노션 개인 칸에 to do list를 작성하고 해야 할 일을 미리 계획함으로써 시간 낭비를 예방하고 해당 기록을 바탕으로 계획적이고 구체적인 개발을 실현하기



선원종

- 서버의 부하 테스트시 응답실패율을 없애고 응답시간을 3초 이내로 줄이기
1. Jmeter를 사용하여 최대 부하가 걸리는 시나리오를 짜보며 커넥션 관리 테스트 진행
  2. 채팅방 로그를 계속해서 저장하는 것이 아닌 채팅방이 종료되게 되면 전체 메시지를 DB에 저장하여 I/O줄이기
  3. 스케일 아웃시 clustering을 미리 구성해두고 그 데이터 공유 문제를 Redis의 Pub/Sub을 매개체로 구현



전혜지

1. DB중복과 무결성 최대한 없이 종속적 규칙을 지키며 설계해보기
2. 쿼리문 작성시 각 절의 순서를 지키고 검색 최소화하여 효율적이고 직관적으로 작성해보기
3. 효율적인 협업을 위해 commit 메시지만을 보고 수정 내용을 알 수 있도록 작성하고, git의 기능 최대한 활용해보기




최성민

1. 서버 호출을 최소화하고 데이터를 효율적으로 읽는 스키마를 설계 (n+1문제, populate, 적절한 내장 모델등을 사용)
2. RESTful API중심의 서버 구성 (URI 표현 규칙 통일, 엄격한 HTTP 상태와 method 사용)
3. MSA 아키텍처를 통한 서버 구성

# 방법 2 : 그라운드 룰 & 데일리 스크럼 회의

**정기 회의**

 (한국) 매주 금요일 밤 10:00 ~  
(외국) 매주 금요일 낮 15:00 ~

New York  
12:03:26 AM  
Tuesday • EST

Los Angeles  
9:03:26 PM  
Monday • PST

South Korea  
2:03:26 PM  
Tuesday • KST

New Zealand  
6:03:26 PM  
Tuesday • NZDT

2023년 2월							오늘 >
일	월	화	수	목	금	토	
29	30	31	2월 1일	2	3	4	
			Daily Scrum 개발사항	Daily Scrum 개발사항	Daily Scrum 개발사항 23.02.03 회의록 회의	Daily Scrum 개발사항	
5	6	7	8	9	10	11	
Daily Scrum 개발사항	Daily Scrum 개발사항	Daily Scrum 개발사항	Daily Scrum 개발사항	Daily Scrum 개발사항	23.02.10 회의록 회의 Daily Scrum 개발사항	Daily Scrum 개발사항	
12	13	14	15	16	17	18	
Daily Scrum 개발사항	Daily Scrum 개발사항	Daily Scrum 개발사항	Daily Scrum 개발사항	대면 회의 및 프로젝트 잡지기 개발사항 23.02.17 회의록 회의			
19	20	21	22	23	24	25	
	Daily Scrum 개발사항	Daily Scrum 개발사항					
26	27	28	3월 1일	2	3	4	

## 팀 규칙

### 1. 코어타임

- 외국 : 최소 두 시간(순코시간) 및 **당일 TodoList 완료**하기
- 한국 : 최소 세 시간(순코시간) 및 **당일 TodoList 완료**하기

### 2. 항상 기록하기 : 당일 개발한 부분은 사소한거라도 **항상 To do List에 기록**하기

### 3. 의사결정 : 결정을 뒷받침할 수 있는 근거를 가져와서 팀원들 간 투표 진행 (잘못되어도 해결방법을 우선으로 생각하고 그것에 대해 불평하지 않기)

### 4. 질문 및 반응

- 질문 사항은 슬랙에 기록하기
- 불만 사항은 카톡방 이용하기
- 읽은경우 잘 모르는 내용이라도 **이모티콘 남기기**
- 질문을렸을경우 카톡방에 올렸다고 **공지**해주기

### 5. 스크럼 & 보고

- 외국
  1. 개발 시작 전, 각자가 개발 할 것에 대해 To do list 작성 후 보고 (노션)
  2. 그 날 개발 완료 후 진행사항에 대해 보고 (노션에 체크표시)
- 한국
  1. 매일 11시 스크럼 회의 (디코)
  2. 개발 시작 전, 각자가 개발 할 것에 대해 To do list 작성 후 보고 (노션)
  3. 스크럼 회의 때 긍정적인 멘트 하나씩 생각하며 팀원들간 격려
  4. 당일 개발 종료 후 보고 (노션)

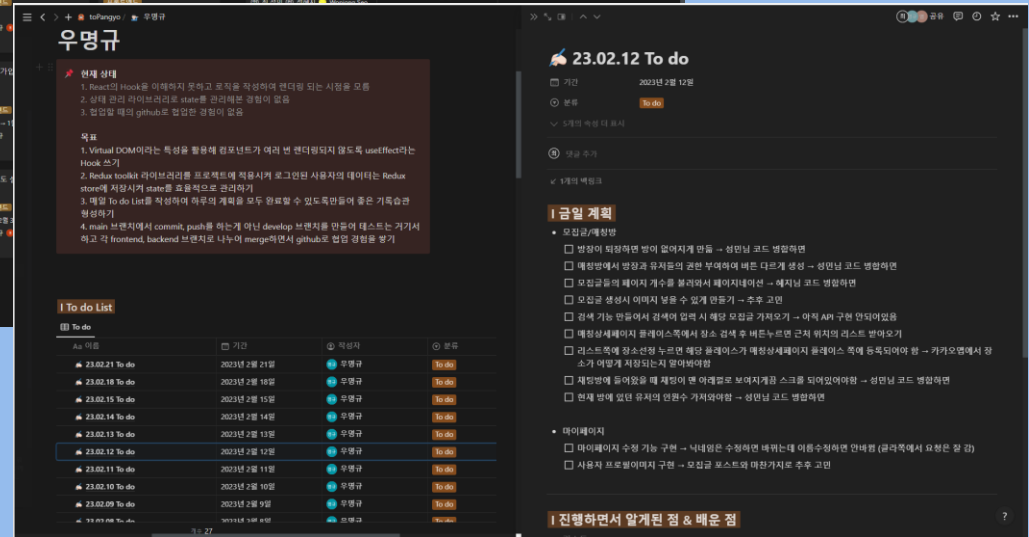
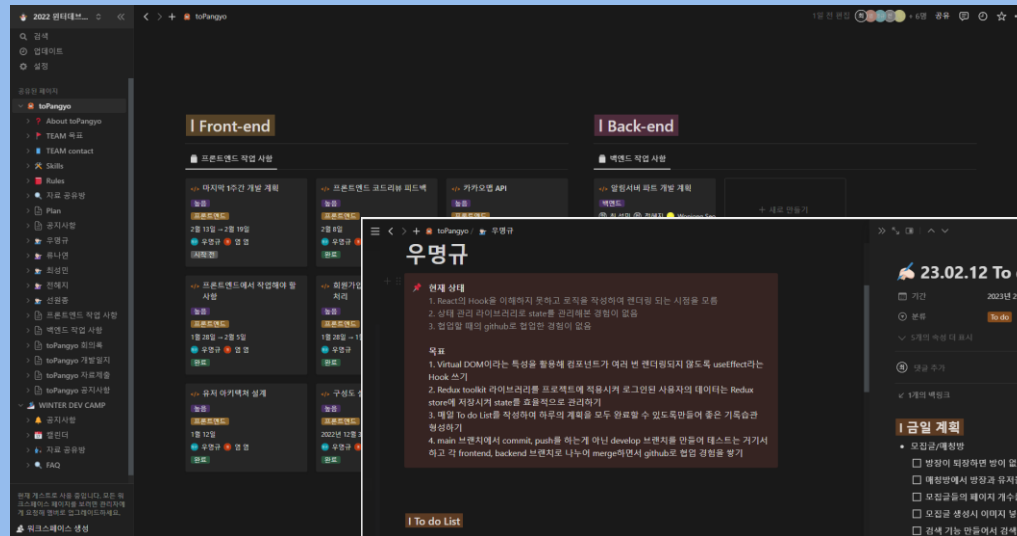
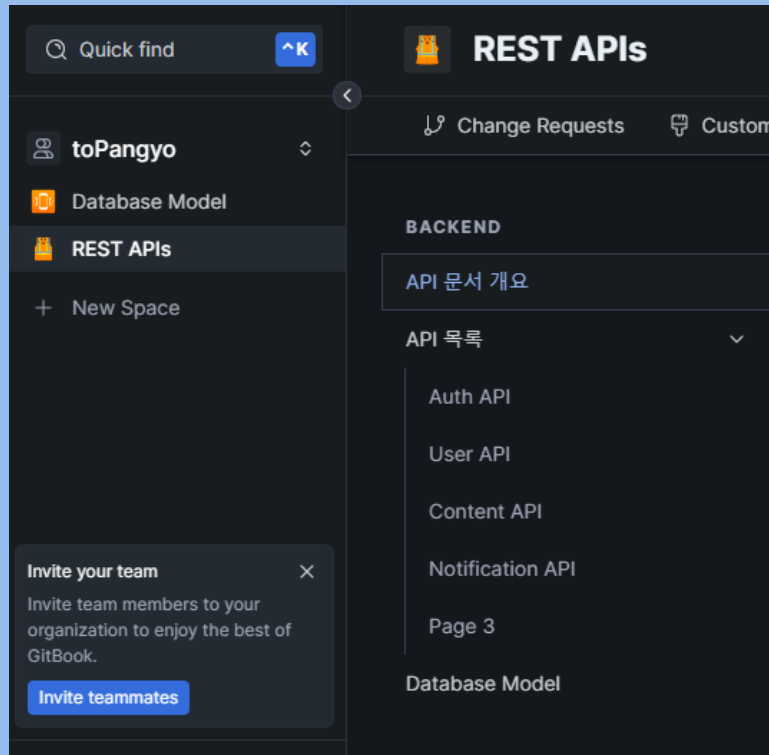
### 6. 이슈 & 문제

- 이슈에 대해 최소 1시간 이상 고민해봤음에도 해결되지 않을 시 **팀원들에게 공유**하기

### 7. 자유시간

- 급하지 않는 한 식사시간엔 **개발**에기 금지
  - 점심 : 12:00 ~ 1:30pm
  - 저녁 : 5:30 ~ 7:00pm

# 방법 2 : 협업 툴




Gitbook

Notion



극복

toPangyo

동세권 

**성공 / 실패 / 부족함**

# 성공 : 클라이언트

## 카카오맵 api를 활용한 구현

- 반경 계산
- 마커 기능 구현
- 현재 위치 받아오기
- Service 라이브러리를 활용한 place 정보 호출 등

## 성능 관련 기능 요소 구현

- 폰트 경량화
- 필요에 따른 스크립트 호출
- 알맞은 hook 사용 등

## 주요 기능의 이벤트 핸들링

- 응답 받은 데이터를 컴포넌트에 바인딩
- 모달을 통한 페이지 호출 줄이기
- 일반적인 CRUD 구현 등

## 협업

- 서버와 RESTful한 데이터 요청 및 응답
- Github를 활용한 관리 및 업데이트 되는 코드

# 성공 : 서버

## 채팅

- 소켓을 이용한 채팅 서버 구현 (채팅, 채팅 방, 장소/시간 선택)
- Room 단위 통제

## 알림

- Redis, FCM, web-push라이브러리, socket등 다양한 방법 이해
- Redis 연결 및 미들웨어로의 사용, DB와 사용하며 I/O성능 개선

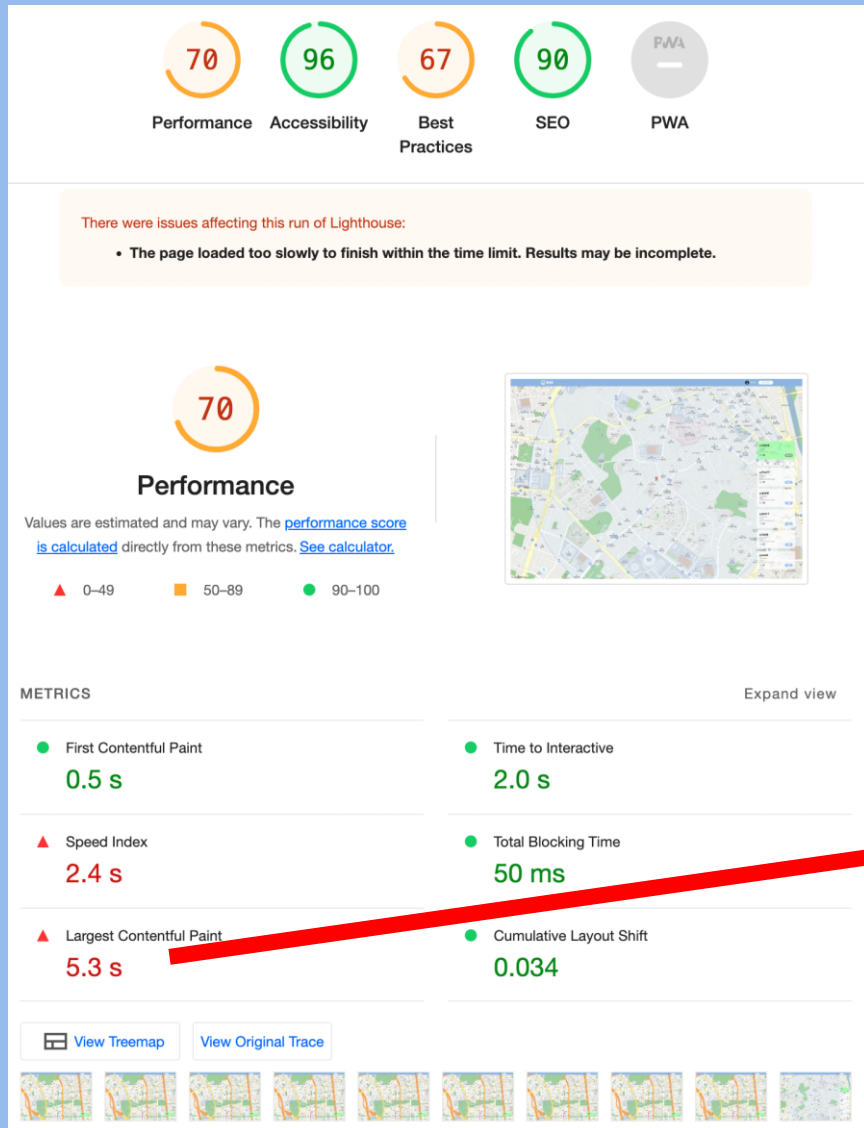
## DB

- RESTful한 api설계
- mySQL 외부 공유 (포트 포워딩)

## 경험

- 아키텍처, 목표설정 등 체계적인 팀 프로젝트의 경험
- Github를 활용한 관리 및 업데이트 되는 코드

# 실패 : LCP



- 페이지가 처음 로드 된 시점을 기준으로, 뷰포트 내에 가장 큰 이미지 또는 텍스트 블록의 렌더링 시간 보고



**LCP : 5.3**

**POOR 레벨**

# 실패 : LCP 원인 분석

## 여러가지 방법

- LCP리소스를 최대한 빠르게 로딩 시작.
- LCP 요소가 로딩을 마친 후 최대한 빠르게 렌더링 되도록 한다.
- LCP 리소스 로드 시간을 줄인다.
- 초기 HTML을 최대한 빠르게 전달한다.

## 적용 가능한 방법

요소 발견 때 최적화

- 필요에 따른 실행 구현

렌더링 지연 제거하기

- 스타일 시트 줄이기 / 스타일 시트 인라인화
- 서버 사이드 렌더링

리소스 로드 시간 줄이기

- 웹 폰트 경량화

# 실패 : LCP 개선기

요소 발견 때 최적화

- 필요에 따른 실행 구현 ☒

(public/index.html)

```
...  
<script  
  type="text/javascript"  
  src="//dapi.kakao.com/v2/maps/sdk.js?appkey=%REACT_APP_KAKAO_MAP%&libraries=services"  
></script>  
...
```



```
...  
const defaultMap = () => {  
  const script = document.createElement("script");  
  script.type = "text/javascript";  
  script.src =  
    "//dapi.kakao.com/v2/maps/sdk.js?appkey=%REACT_APP_KAKAO_MAP%&autoload=false";  
  script.async = true;  
  document.head.appendChild(script);  
  script.onload = () => {  
    window.kakao.maps.load(() => {  
      const container = document.getElementById("map");  
      var options = {  
        //지도를 생성할 때 필요한 기본 옵션  
        center: new window.kakao.maps.LatLng(37.365264512305174, 127.10676860117488), //지도의 중심좌표.  
        level: 3, //지도의 레벨(확대, 축소 정도)  
      };  
      var kakaoMap = new window.kakao.maps.Map(container, options);  
      dispatch(setMap(kakaoMap));  
    })  
  }  
}  
...  
useEffect(() => {  
  defaultMap()  
}, [1])
```

원인 1)

용량이 큰 불필요한 요소의 항상 실행

Index 실행에 따라 카카오맵api가 처음부터 렌더링 됨.

변경 1)

필요에 따른 가져오기로 변환.

해당 페이지가 렌더링 될 때 script 생성

# 실패 : LCP 개선기

## 적용 가능한 방법

렌더링 지연 제거하기

- 스타일 시트 줄이기

/ 스타일 시트 인라인화 ☒

- 서버 사이드 렌더링 ✕

리소스 로드 타임 줄이기

- 웹 폰트 경량화 ☒

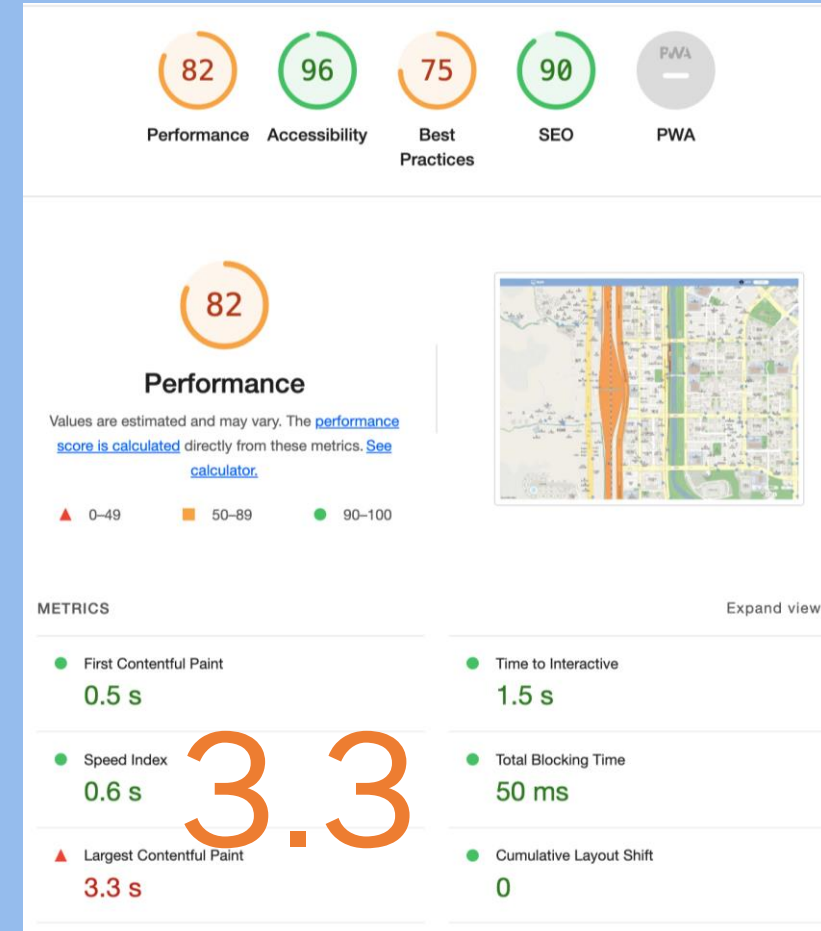
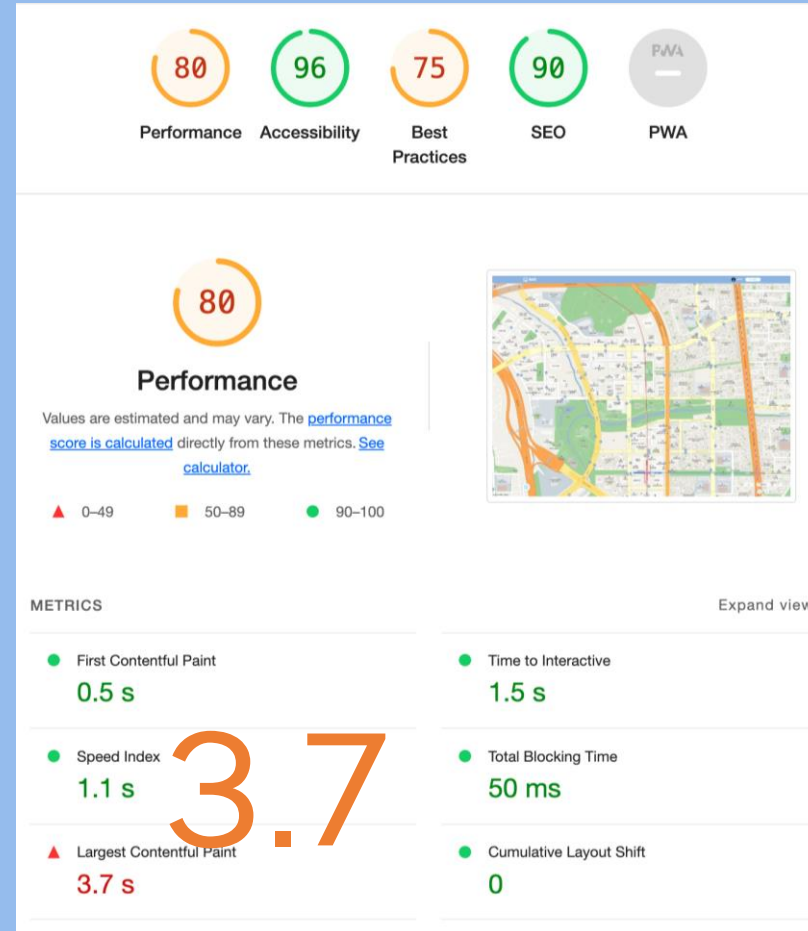
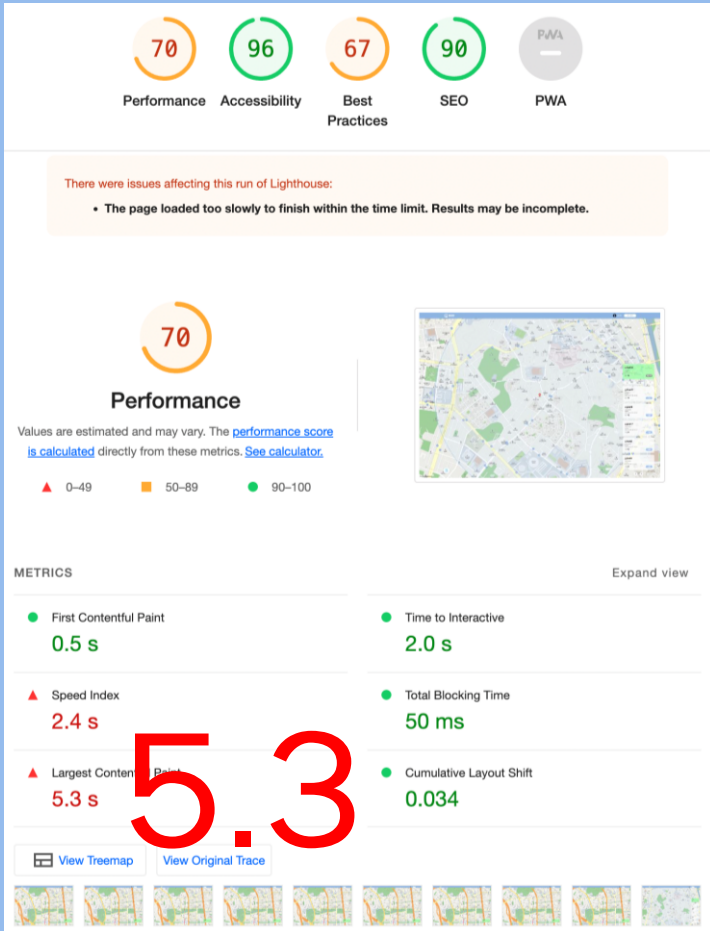
```
@font-face {
  font-family: 'normal';
  font-weight: 500;
  font-style: normal;
  src: url('https://cdn.jsdelivr.net/gh/webfontworld/pretendard/Pretendard-Medium.woff2') format('woff2'),
        url('https://cdn.jsdelivr.net/gh/webfontworld/pretendard/Pretendard-Medium.woff') format('woff'),
        url('https://cdn.jsdelivr.net/gh/webfontworld/pretendard/Pretendard-Medium.ttf') format("truetype");
  font-display: swap;
}

@font-face {
  font-family: 'bold';
  font-weight: 800;
  font-style: normal;
  src: url('https://cdn.jsdelivr.net/gh/webfontworld/pretendard/Pretendard-ExtraBold.woff2') format('woff2'),
        url('https://cdn.jsdelivr.net/gh/webfontworld/pretendard/Pretendard-ExtraBold.woff') format('woff'),
        url('https://cdn.jsdelivr.net/gh/webfontworld/pretendard/Pretendard-ExtraBold.ttf') format("truetype");
  font-display: swap;
}

@font-face {
  font-family: 'title';
  src: url('https://cdn.jsdelivr.net/gh/projectnoonnu/noonfonts_five@.2.0/UhBeeSe_hyun.woff') format('woff');
  font-weight: normal;
  font-style: normal;
}
```



# 실패 : 이후..



# 실패, 부분성공 : 클라이언트

## 코드

- 효율적인 코드 (ex 코드 중복)
- 예외처리
- 성능 최적화
- 반응형 웹 구현

## 무한 스크롤 구현

## 페이지네이션

## 이벤트 다루기 (일부 인원)

- 리액트를 처음 접하면서 UI를 제작하는 것에 집중

## RTK활용 미흡과 prop drilling 문제

- 페이지 설계 단계에서 전역으로 사용될 변수 및 root props등을 파악한다.

## 컴포넌트에서의 실시간 데이터 전달

- socket을 전역으로 둔다

## 무분별한 axios호출

- 파일을 분리해서 관리한다.

## SCSS활용 미비

- 실력적 원인

## 계획했던 라이브러리 활용 못함

- 활용하기에는 다른 부분에서 할 게 많음

# 실패, 부분성공 : 서버

## 모델 인덱싱

- 필요 속성 호출로 대체, 사전 고려 요소가 불충분 했고 설계 과정에서 다시 확인하기.

## 채팅 클러스터링

- 싱글 구현을 우선 진행

## 배포

- AWS 등 시간과 역량 문제가 원인

## Redis pub/sub만을 이용한 알림 구현

- 단독 알림 서버 구현 실패, 채팅과 비슷하다고 생각하여 추후 채팅과 비교하여 분석

## 중복 없는 DB작성

- 서버 별로 DB를 나누다 보니 발생 설계 문제

## MSA 아키텍처를 통한 서버 구성

- 통합 되었으면 좋겠다 싶은 부분이 많았음. 설계 기준을 다시 확인하고 리팩토링

## 서버 호출 최소화

- 코드 리뷰를 받으며 리팩토링

## Docker 컨테이너화

- 컨테이너화의 부재를 느낌, 역량 문제

## Task queue 도입

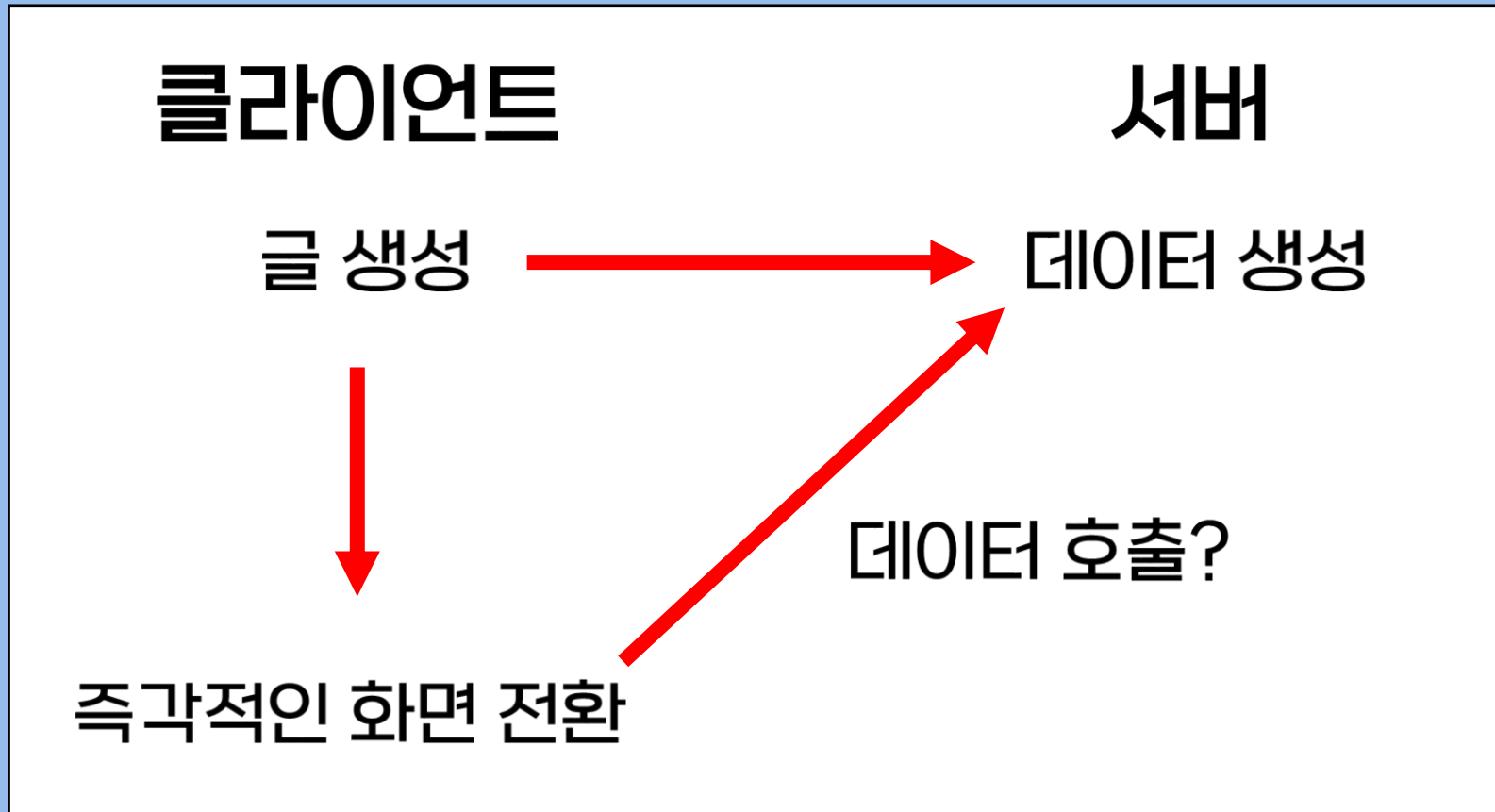
- 안정적인 서비스를 위한 도입을 시도 못함. 역량 문제.

## 복잡한 코드

- 너무 여러 방법을 찾아보았으며 시간 관리 미흡

# 실패, 부분성공 : 클라이언트 & 서버

데이터베이스 저장과 호출 시간의 겹침 문제



# 실패, 부분성공 : 설계, 잘 좀 할 걸

$A \rightarrow B \rightarrow C$ 로  
해야지

클라이언트

$A \rightarrow B$ 면  
되겠지?

서버

# 주요 원인 1 : 역량 파악 실패

## 12월, 초기에 작성한 기능

### 기능

#### ▼ 🚧 유저

- 🚧 로그인/회원가입
  - 로그인 → jwt 토큰 갱신 api path
  - 🚧 OAuth 나중엔 코드 바로 들어갈 수 있도록 예상하여 설계하기
- 🚧 마이페이지
  - 🚧 회원정보 수정 및 옵션 설정, 🚧 글 작성 내역(매칭 여부)
- 🚧 알림
- 🚧 신고
- 🚧 친구
- 🚧 인기도 (매너온도)
- 🚧 육성기능 (레벨,칭호,아바타, 포인트 etc)

#### ▼ 🚧 매칭

- 🚧 현재 위치 기반 글 리스트 / 재검색
  - 포스트에서 위도 경도 저장 하고 개별 사용자가 내 위치 기준 포스트와의 거리 비교 해서 리스트 출력
- 🚧 카테고리 분류
- 🚧 검색

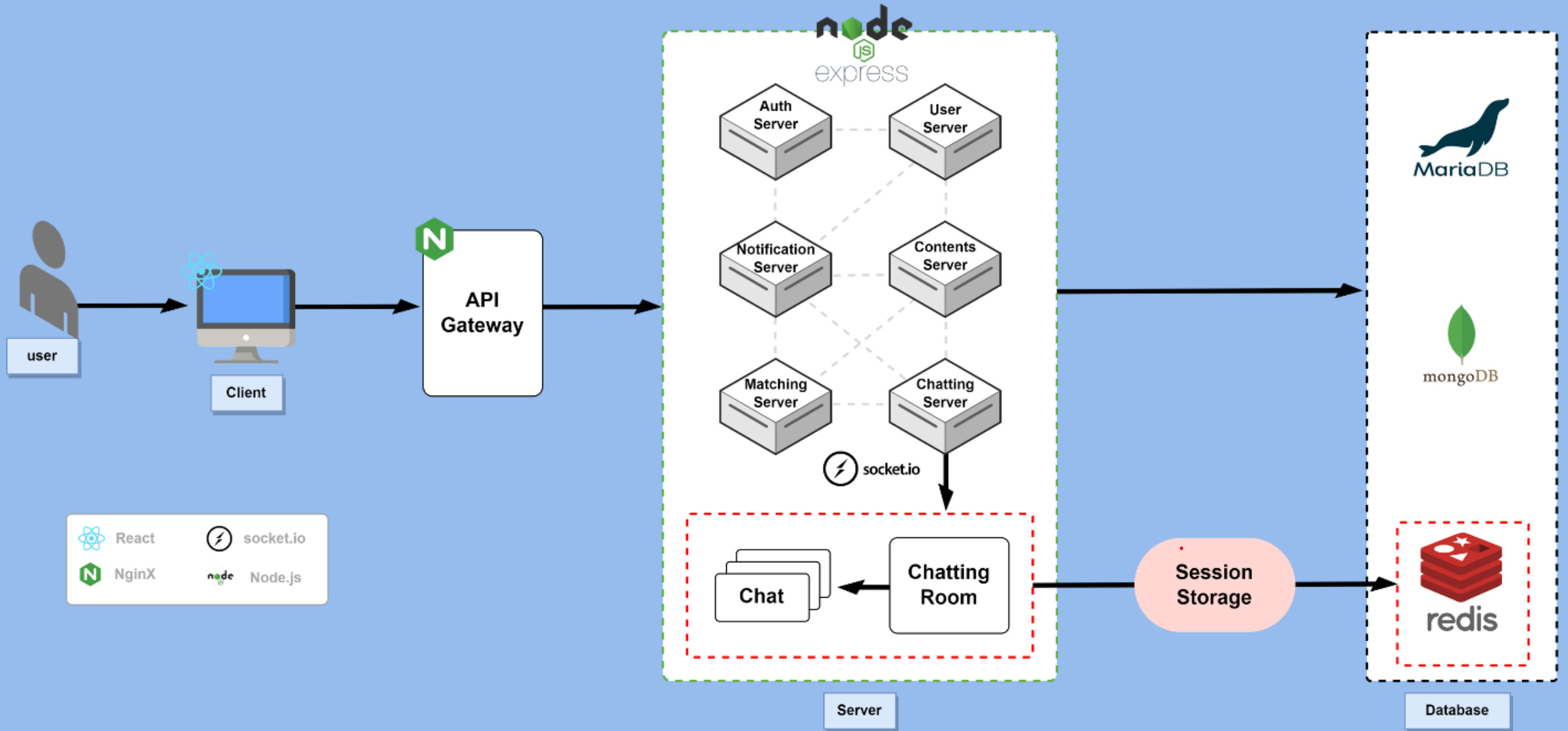
#### ▼ 🚧 개별 포스트(모집)

- 공용
  - 🚧 방장 권한 (강퇴)
  - 🚧 문자 채팅
  - 🚧 시간 설정(본문)
  - 🚧 음성 채팅
- 확정 전
  - 🚧 인원 설정
  - 🚧 나가기 (사유 선택)
  - 🚧 확정여부 → 신청 마감
  - 🚧 신청 목록(수락/거절)
- 확정 후
  - 🚧 시간 및 장소 설정
  - 🚧 장소 추천
  - 🚧 장소 선정 후 플레이스 상세 정보 알려주기.
- 만남 이후
  - 🚧 매칭 성사 여부
  - 🚧 유저 평가
  - 🚧 플레이스 평가 - 별점, 한줄평

#### ▼ 🚧 큐레이팅

- 게시판CRUD

# 주요 원인 1 : 역량 파악 실패



## 주요 원인 2 : 시간 관리 실패

Time is **GOLD**



# 후기

Know yourself

**“너 자신을 알라!”**

**주제 파악 좀 해라**

# 여담 : aws

aws

서비스

?

🔍

📧

🔔

🔗

글로벌 ▼

junnsong ▼

홈

결제

청구서

결제

크레딧

구매 주문

Cost & usage reports

Cost categories

비용 할당 태그

Free tier

Billing

Conductor

Cost Management

Cost explorer

Budgets

Budgets reports

절감 계획

기본 설정

결제 기본 설정

결제 기본 설정

통합 결제

세금 설정

권한

Affected policies

청구서

새로운 청구서 페이지 환경을 사용할 수 있습니다.  
더 쉽게 사용할 수 있도록 청구서 페이지를 재설계했습니다. 새로운 환경을 사용해 보십시오.

날짜: 2월 2023

CSV 다운로드

인쇄

예상 총액

527,261 KRW

\$411.24

인보이스 총액은 인보이스가 발행되면 표시됩니다.

세부 정보

AWS 서비스 요금

\$411.24

CloudWatch

\$0.00

Key Management Service

\$0.00

Relational Database Service

\$373.85

Simple Storage Service

\$0.00

세금

납부할 VAT

\$37.39

본 청구서 기간 동안의 사용 요금 및 반복 청구 요금은 다음 결제 일자에 청구됩니다. 이 페이지 또는 귀하에게 전송된 알림에 표시된 예상 요금은 본 청구서 기간 동안의 실제 요금과 다를 수 있습니다. 이 페이지에 나와 있는 예상 요금에는 귀하가 이 페이지를 확인한 후 본 청구서 기간 동안 발생한 사용 요금은 포함되어 있지 않기 때문입니다. 또한 알림을 통해 귀하에게 제공된 예상 요금 정보에는 알림을 보낸 이후 본 청구서 기간 동안 발생한 사용 요금은 포함되어 있지 않습니다. 월회성 요금 및 구독 요금은 사용 요금 및 반복 청구 요금과는 별도로 발생일을 기준으로 산정됩니다. 이 페이지의 요금은 별도 항목으로 나열되어 있지 않는 한 세금을 제외한 금액입니다. 세금 정보에 액세스하려면 단종 AWS Organization 관리 책임자에게 문의하십시오.

인보이스 결제 시 미국 달러가 아닌 다른 통화를 선택했습니다. 다른 통화로 결제하려면 계정 설정 페이지에서 기본 설정을 업데이트하십시오. Amazon Services LLC가 통화를 변환합니다.

의견

언어

개인 정보 보호

약관

쿠키 기본 설정

© 2023, Amazon Web Services, Inc. 또는 계열사.

aws

서비스

🔍

📧

🔔

더 보기 ▼

대금 및 비용 관리

AWS 결제 대시보드

이번 달의 총 예상

USD 1,626.25

KRW 2,085,057.85

현재 당월 누계 잔액

USD 411.24

KRW 527,260.80

추세가 있는 동일한 기간의 이전 달

표시할 데이터 없음

총 활성 서비스 수

4

의견

언어

개인 정보 보호

약관

쿠키 기본 설정

© 2023, Amazon Web Services, Inc. 또는 계열사.

# 후기

## 우명규

약 세 달간 프로젝트를 진행하면서 부족한 부분과 배워야하는 부분을 확인할 수 있는 좋은 경험이 된 것 같았고 프론트엔드 개발자라는 목표에 더 다가갈 수 있었던 것 같습니다. 팀원들 모두 프로젝트 하느라 고생 많았고 덕분에 잘 헤쳐 나가서 프로젝트를 마칠 수 있었습니다. 앞으로 다들 멋진 개발자가 돼서 또 함께 발전된 프로젝트를 할 수 있으면 좋겠습니다 :)

## 류나연

좋은 사람들과 양질의 경험을 함께 할 수 있어 뜻깊은 시간이었습니다. 좋은 개발자가 되기 위해 어떻게 나아가야 할지 고민해보는 나날들이었다고 생각합니다. 이런 기회를 얻을 수 있어 참 운이 좋았던 것 같습니다. 취직까지도 이어져 모두 운명적으로 판교에서 만나면 좋겠습니다.

## 선원종

이번 데브 캠프의 경험을 통해서 앞으로 개발이 아닌 어떠한 분야의 프로젝트를 담당해도 당당하게 말을 수 있게 되었습니다. 체계적인 일정관리와 사전계획의 중요성을 깨달았습니다. 또 무엇보다 좋은 팀원들을 만나서 팀원들에게 너무 감사하고 앞으로도 좋은 인연을 이어나갈 수 있으면 좋겠습니다.

## 최성민

프로젝트를 진행하며 이런 저런 정말 많은 시행착오를 겪고 정말 많은 성공, 실패를 겪었습니다. 특히, 이 과정을 함께 공유한 팀원분들이 있어서 더욱 뜻깊었습니다. 처음 계획한 구현을 완성하지 못한 것은 아쉬우나 그 덕분에 더 값진 경험을 가져갈 수 있었습니다. 성공을 위해 함께 몰두하고 완주하는 이 경험, 감사합니다. 팀 이름대로 모두 판교에서 만납시다.

## 전혜지

계획을 세우고, db 설계, api 명세서 작성 등 체계적으로 프로젝트를 한 것이 처음이라 경험 자체가 값지고 소중했습니다. 처음 시도하는 것들이 많아 계속적으로 한계에 부딪치고, 큰일이 날 뻔도 했는데 도움 요청할 때마다 디코로 달려와 도와 주시고 배려해 주신 모든 팀원분들께 감사하다는 말밖에는 못 하겠네요. 끝까지 다 같이 완주해서 기쁘고, 모두 판교에서 봐요!

**최고의 팀**

toPangyo

