REACT

에어 코리아 API - 대기 오염 정보 조회 서비스

작성자 :최수진 박세영

목 차

- API 소개 및 링크
- API 사용 방법
- 실행 결과
- 소스 코드
- 소감

API 소개 및 링크

- 소개
  - 각 측정소별 대기오염정보를 조회하기 위한 서비스 입니다.
- 사용 가능한 기능
  - 측정소별 실시간 측정정보 조회
  - 통합대기환경지수 나쁨 이상 측정소 목록조회
  - 시도별 실시간 측정정보 조회
  - 대기질 예보통보 조회
  - 초미세먼지 주간예보 조회
- 링크 - 공공 데이터 포털
  - https://www.data.go.kr/index.

- 이 API를 선택한 이유

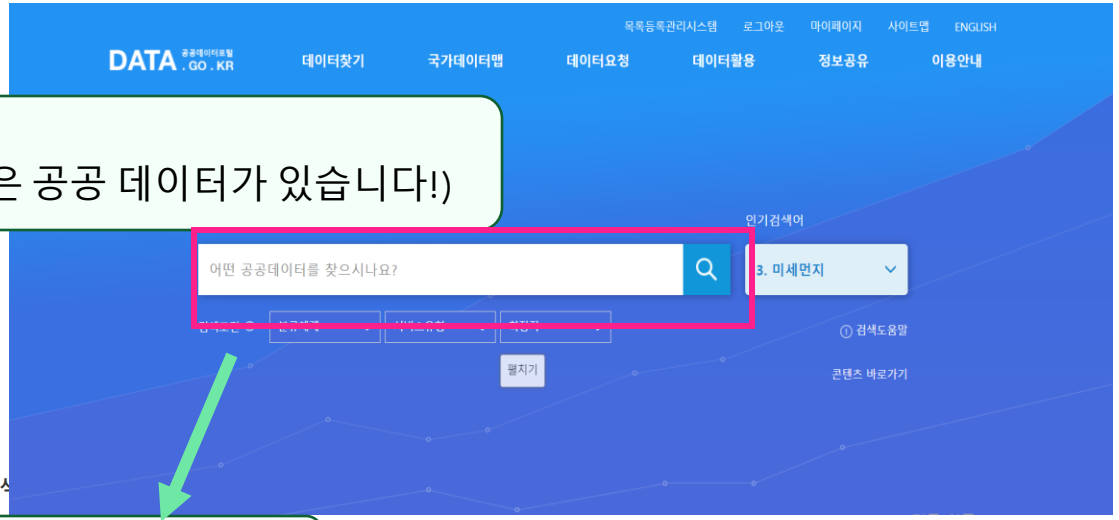
  코로나의 영향력이 약해져 마스크를 밖에서 벗을 수 있게되었지만 미세먼지는 아직 현재 진행중입니다.
  어쩌면 평생 미세먼지를 걱정해야할지도 모릅니다.
  코로나로 인해 2순위로 밀려났던 미세먼지 걱정이 이제는 1순위로 바뀔수있고
  미세먼지로 인해 마스크를 계속 착용을 할수도 있기에
  현재 저희가 살고있는 지역의 대기질에 자연스레 관심을 갖게되어 선택하게 되었습니다.

API 사용 방법

- KEY 발급 받기

1. 공공 데이터 포털 홈페이지 접속
https://www.data.go.kr/index.do

2. 검색창에 '에어 코리아' 검색
(저희가 선택한 API외에 정말 많은 공공 데이터가 있습니다!)



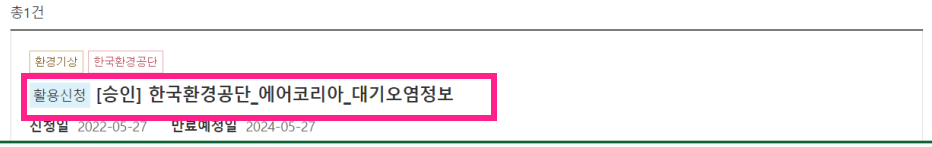
3. 오픈 API 선택

API 사용 방법

- KEY 발급 받기

4. 한국환경공단_에어코리아_대기오염정보에서 활용신청 클릭!

5. 활용 목적 선택과 내용 기제후 사용할 상세 기능 정보 선택 (기본적으로 전부 선택되어 있습니다.)



6. 발급이 완료되면 마이 페이지로 넘어와 활용 신청이 완료된것을 확인할수 있습니다.

API 사용 방법

신청 완료후 클릭시 자세한 정보를 확인할수 있습니다.

서비스정보

참고문서	에어코리아_대기오염정보 조회 서비스_기술문서_v1.0.docx
데이터포맷	JSON+XML
End Point	http://apis.data.go.kr/B552584/ArpltnInforInqireSvc
API 환경 또는 API 호출 포털에서 제공되는 Encc * 향후 포털에서 더 명확	
일반 인증키 (Encoding)	3D
일반 인증키 (Decoding)	aG5MSosECAZ12G4HhKWz/AlBA7PZ6QvpeFSUk4x4vIBidPv/ajaKl1H8jc9GcQj0YdHIMdxaVzZQ0e63O416tw==

또한 활용 신청 상세 기능 정보를 확인할수 있는데
미리보기 항목의 확인 버튼 클릭시 샘플 데이터를 입력해
직접 확인할수 있습니다.

활용신청 상세기능정보

NO	상세기능	설명	일일 트래픽	미리보기
1	대기질 예보통보 조회	통보코드와 통보시간으로 예보정보와 발생 원인 정보를 조회하는 대기질(미세먼지/오존) 예보통보 조회	500	확인
2	초미세먼지 주간예보 조회	통보코드와 통보시간으로 대기질 전망과 주간예보 정보를 조회하는 초미세먼지 주간예보통보 조회	500	확인
3	측정소별 실시간 측정정보 조회	측정소명과 측정데이터 기간(일,한달,3개월)으로 해당 측정소의 일반항목 측정정보를 제공하는 측정소별 실시간 측정정보조회	500	확인
4	통합대기환경지수 나쁨 이상 측정소 목록조회	통합대기환경지수가 나쁨 등급 이상인 측정소명과 주소 목록 정보를 제공하는 통합대기환경지수 나쁨 이상 측정소 목록조회	500	확인
5	시도별 실시간 측정정보 조회	시도명을 검색조건으로 하여 시도별 측정소목록에 대한 일반 항목과 CAI최종 실시간 측정값과 지수 정보 조회 기능을 제공하는 시도별 실시간 측정정보 조회	500	확인

NO	상세기능	설명	일일 트래픽	미리보기
1	대기질 예보통보 조회	통보코드와 통보시간으로 예보정보와 발생 원인 정보를 조회하는 대기질(미세먼지/오존) 예보통보 조회	500	확인

요청변수(Request Parameter) 닫기

항목명	샘플데이터	설명
serviceKey	-	공공데이터포털에서 받은 인증키
returnType	xml	xml 또는 json
numOfRows	100	한 페이지 결과 수(조회 날짜로 검색 시 사용 안함)
pageNo	1	페이지번호(조회 날짜로 검색 시 사용 안함)
searchDate	2020-11-14	통보시간 검색(조회 날짜 입력이 없을 경우 한달동안 예보통보 발령 날짜의 리스트 정보를 확인)
InformCode	PM10	통보코드검색(PM10, PM25, O3)

미리보기

미리보기 예시

This XML file does not appear to have any style

```
▼<response>
  ▼<header>
    <resultCode>00</resultCode>
    <resultMsg>NORMAL_CODE</resultMsg>
  </header>
  ▼<body>
    <items/>
    <numOfRows>100</numOfRows>
    <pageNo>1</pageNo>
    <totalCount>0</totalCount>
  </body>
</response>
```

xml

json

{"response":{"body":{"totalCount":40,"items":
[{"so2Grade":"1","coFlag":null,"khaiValue":"63","so2Value":"0.004","coValue":"0.3","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.020","stationName":"중구","pm10Grade":"1","o3Value":"0.046"},
{"so2Grade":"1","coFlag":null,"khaiValue":"57","so2Value":"0.003","coValue":"0.5","pm25Flag":null,"pm10Flag":null,"o3Grade":"1","pm10Value":"41","khaiGrade":"2","pm25Value":"18","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.033","stationName":"한강대로","pm10Grade":"2","o3Value":"0.024"},
{"so2Grade":"1","coFlag":null,"khaiValue":"62","so2Value":"0.003","coValue":"0.4","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"25","khaiGrade":"2","pm25Value":"16","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.021","stationName":"중로구","pm10Grade":"1","o3Value":"0.044"},
{"so2Grade":"1","coFlag":null,"khaiValue":"66","so2Value":"0.003","coValue":"0.4","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"27","khaiGrade":"2","pm25Value":"10","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.024","stationName":"청계천로","pm10Grade":"1","o3Value":"0.037"},
{"so2Grade":"1","coFlag":null,"khaiValue":"53","so2Value":"0.003","coValue":"0.5","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"26","khaiGrade":"2","pm25Value":"15","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.028","stationName":"종로","pm10Grade":"1","o3Value":"0.033"},
{"so2Grade":"1","coFlag":null,"khaiValue":"70","so2Value":"0.003","coValue":"0.3","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"18","khaiGrade":"2","pm25Value":"7","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.017","stationName":"용산구","pm10Grade":"1","o3Value":"0.054"},
{"so2Grade":"1","coFlag":null,"khaiValue":"59","so2Value":"0.002","coValue":"0.3","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"22","khaiGrade":"2","pm25Value":"9","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.016","stationName":"광진구","pm10Grade":"1","o3Value":"0.041"},
{"so2Grade":"1","coFlag":null,"khaiValue":"52","so2Value":"0.003","coValue":"0.3","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"20","khaiGrade":"2","pm25Value":"10","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.021","stationName":"성동구","pm10Grade":"1","o3Value":"0.044"},
{"so2Grade":"1","coFlag":null,"khaiValue":"56","so2Value":"0.002","coValue":"0.7","pm25Flag":null,"pm10Flag":null,"o3Grade":"1","pm10Value":"23","khaiGrade":"2","pm25Value":"8","sidoName":"서
울","no2Flag":null,"no2Grade":"2","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.034","stationName":"강변북로","pm10Grade":"1","o3Value":"0.028"},
{"so2Grade":"1","coFlag":null,"khaiValue":"51","so2Value":"0.003","coValue":"0.4","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"25","khaiGrade":"2","pm25Value":"12","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.021","stationName":"중랑구","pm10Grade":"1","o3Value":"0.043"},
{"so2Grade":"1","coFlag":null,"khaiValue":"63","so2Value":"0.003","coValue":"0.3","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"25","khaiGrade":"2","pm25Value":"9","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.021","stationName":"종대문구","pm10Grade":"1","o3Value":"0.046"},
{"so2Grade":"1","coFlag":null,"khaiValue":"55","so2Value":"0.003","coValue":"0.5","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"28","khaiGrade":"2","pm25Value":"14","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.029","stationName":"홍릉로","pm10Grade":"1","o3Value":"0.036"},
{"so2Grade":"1","coFlag":null,"khaiValue":"56","so2Value":"0.003","coValue":"0.3","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"22","khaiGrade":"2","pm25Value":"15","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-27 20:00","coGrade":"1","no2Value":"0.022","stationName":"성북구","pm10Grade":"1","o3Value":"0.037"},

API 사용 방법

- 자료 사항 시 주의 사항은 기술문서 안과 공공데이터 홈페이지의 마이 페이지안의 활용신청이 완료된 API 클릭 시 맨 밑의 하단에서 확인 할 수 있습니다.

3.2 자료 이용 시 준수사항

(1) API 활용 신청서의 "활용용도" 이외에는 사용을 제한하며 반드시 자료의 출처(환경부/한국환경공단)표기 의무를 준수하여야 함(약관 제 7 조)

(2) 우리 기관이 제공하는 자료는 "인증을 받지 않은 실시간자료"이므로 자료 오류 및 표출방식에 따라 값이 다를 수 있음을 명시해야 함(약관 제 8 조)

라이센스표시

이용허락범위	저작자표시-변경금지 (사유 :'출처 및 데이터 오류가능성 표시')

- 항목중 'Grade'라는 단어가 포함된 값들이 많습니다. 'Grade'는 항목에 대한 등급을 표시한 값입니다.

※ 항목별 Grade 값의 의미

- 적용 항목명 : khaiGrade, so2Grade, coGrade, o3Grade, no2Grade, pm10Grade, pm25Grade, pm25Grade1h, pm25Grade1h

등급	좋음	보통	나쁨	매우나쁨
Grade 값	1	2	3	4

- 실제 저희가 받아온 데이터의 스크린샷 입니다.

localhost:3003

openAPI

data안에 response안에 body안에 items정보들

[{"so2Grade":"1","coFlag":null,"khaiValue":"66","so2Value":"0.004","coValue":"0.3","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"26","khaiGrade":"2","pm25Value":"5","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.013","stationName":"중구","pm10Grade":"1","o3Value":"0.049"},
{"so2Grade":"1","coFlag":null,"khaiValue":"70","so2Value":"0.003","coValue":"0.5","pm25Flag":null,"pm10Flag":null,"o3Grade":"1","pm10Value":"60","khaiGrade":"2","pm25Value":"14","sidoName":"서
울","no2Flag":null,"no2Grade":"2","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.035","stationName":"한강대로","pm10Grade":"2","o3Value":"0.022"},
{"so2Grade":"1","coFlag":null,"khaiValue":"66","so2Value":"0.003","coValue":"0.3","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"35","khaiGrade":"2","pm25Value":"5","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.012","stationName":"종로구","pm10Grade":"1","o3Value":"0.049"},
{"so2Grade":"1","coFlag":null,"khaiValue":"58","so2Value":"0.003","coValue":"0.5","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"29","khaiGrade":"2","pm25Value":"10","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.018","stationName":"청계천로","pm10Grade":"1","o3Value":"0.040"},
{"so2Grade":"1","coFlag":null,"khaiValue":"58","so2Value":"0.003","coValue":"0.4","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"52","khaiGrade":"2","pm25Value":"8","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.020","stationName":"종로","pm10Grade":"2","o3Value":"0.038"},
{"so2Grade":"1","coFlag":null,"khaiValue":"78","so2Value":"0.003","coValue":"0.2","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"36","khaiGrade":"2","pm25Value":"7","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.009","stationName":"용산구","pm10Grade":"2","o3Value":"0.063"},{"so2G
","so2Value":"0.003","coValue":"0.2","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"35","khaiGrade":null,"pm25Value":"3","sidoName":"서울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade"
17:00","coGrade":"1","no2Value":"0.010","stationName":"광진구","pm10Grade":null,"o3Value":"0.040"},
{"so2Grade":"1","coFlag":null,"khaiValue":"71","so2Value":"0.003","coValue":"0.2","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"32","khaiGrade":"2","pm25Value":"2","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.012","stationName":"성동구","pm10Grade":"1","o3Value":"0.055"},
{"so2Grade":"1","coFlag":null,"khaiValue":"57","so2Value":"0.003","coValue":"0.5","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"42","khaiGrade":"2","pm25Value":"5","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.024","stationName":"강변북로","pm10Grade":"2","o3Value":"0.036"},
{"so2Grade":"1","coFlag":null,"khaiValue":"68","so2Value":"0.003","coValue":"0.2","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"25","khaiGrade":"2","pm25Value":"4","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.011","stationName":"중랑구","pm10Grade":"1","o3Value":"0.051"},
{"so2Grade":"1","coFlag":null,"khaiValue":"69","so2Value":"0.003","coValue":"0.2","pm25Flag":null,"pm10Flag":null,"o3Grade":"2","pm10Value":"33","khaiGrade":"2","pm25Value":"5","sidoName":"서
울","no2Flag":null,"no2Grade":"1","o3Flag":null,"pm25Grade":"1","so2Flag":null,"dataTime":"2022-05-26 17:00","coGrade":"1","no2Value":"0.012","stationName":"동대문구","pm10Grade":"1","o3Value":"0.053"},

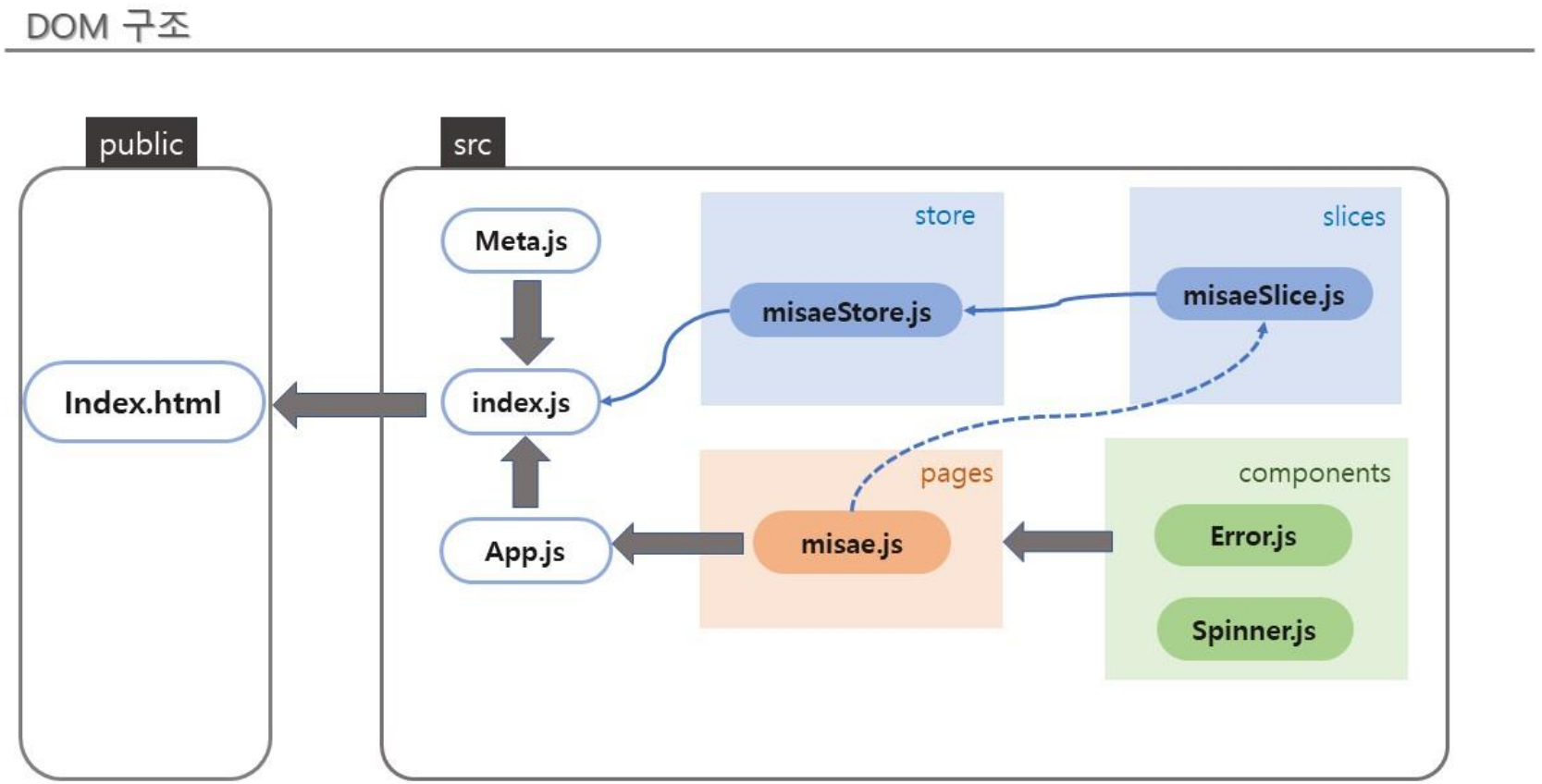
최종 구현 결과

첫 화면

자치구 선택시 화면

로딩시 화면

구현 결과- DOM구조


DOM 구조
public
src
Index.html
Meta.js
index.js
App.js
store
misaeStore.js
slices
misaeSlice.js
pages
misae.js
components
Error.js
Spinner.js

초기 구현 결과

앱처럼 구현하고 싶어 최대한 앱처럼 구성을 했습니다.

초기 디자인

openAPI

종로구

서울특별시

종로구

보통

- 미세먼지: 19㎍/㎥😀
- 초미세먼지: 7㎍/㎥😀

아황산가스 좋음😀 0.003
일산화탄소 좋음😀 0.3
오존 보통😐 0.051
이산화질소 좋음😀 0.011

참고한 디자인
출처:: https://velog.io/@jieun0915/%EB%AF%B8%EC%84%B8%EB%A8%BC%EC%A7%80-%EC%95%B1

소스 코드

초기 소스 코드입니다

src/App.js

```
/**
 * @filename: App.js
 * @description: index.js에 페이지 라우팅
 * @author: 최수진(sujin971008@gmail.com)
 */
import React from "react";

import Misea from "./pages/misea";

function App() {
  return (
    <div>
      <Misea />
    </div>
  );
}

export default App;
```

src/index.js

```
/**
 * @filename: index.js
 * @description: store로 부터 구독받아서 App.js를 통해서 전달
 * @author: 최수진(sujin971008@gmail.com)
 */

import React from "react";
import ReactDOM from "react-dom/client";
import App from "./App";
import { BrowserRouter } from "react-router-dom";
import Meta from "./Meta";

/* 리덕스 구성을 위한 참조 */
import { Provider } from "react-redux";
import store from "./store/misaeStore";

const root = ReactDOM.createRoot(document.getElementById("root"));
```

```
root.render(
  <React.StrictMode>
    <Meta />
    <Provider store={store}>
      <BrowserRouter>
        <App />
      </BrowserRouter>
    </Provider>
  </React.StrictMode>
);
```

src/Meta.js

```
import React from "react";
import { Helmet, HelmetProvider } from "react-helmet-async";

const Meta = (props) => {
  return (
    <HelmetProvider>
      <Helmet>
        <meta charset="utf-8" />
        <title>{props.title}</title>
        {/* SEO 태그 */}
        <meta name="description" content={props.description} />
        <meta name="keywords" content={props.keywords} />
        <meta name="author" content={props.author} />
        <meta property="og:type" content="website" />
        <meta property="og:title" content={props.title} />
        <meta property="og:description" content={props.description} />
        <meta property="og:image" content={props.image} />
        <meta property="og:url" content={props.url} />
        <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />

        {/* Helmet 안에서 CSS적용하기 */}
        <style type="text/css">{`
            *{
                list-style: none;
            }

            body {
                margin: 0;
                padding: 0;
            }

            `}</style>
        {/* 추가적으로 적용해야 할 외부 js나 css로 여기서 명시할 수 있다. */}
      </Helmet>
    </HelmetProvider>
  );
```

```
};
/* 이게 props */
Meta.defaultProps = {
  title: "useage of OpenAPI",
  description: "OpenAPI 활용 제출물입니다.",
  keywords: "OpenAPI",
  author: "최수진,박세영",
  url: window.location.href,
};

export default Meta;
```

src/slices/misaeSlice.js

```
/**
 * @filename: misaeSlice.js
 * @description: axios처리 및 상태값 관리하는 slice.js
 * @author: 최수진(sujin971008@gmail.com)
 */

import { createSlice, createAsyncThunk } from "@reduxjs/toolkit";
import axios from "axios";

/* KEY */
const KEY =

"H1klrl5LXfzjFdnp01Zfej7sKKi5WysCM9D5tQRb0tp3pOXckKcmZnf%2FG3Gm8ESgORTZfC6QCMleUYP
p%2F5MHaw%3D%3D";

/* 비동기 처리 함수 구현 */
//  payload는 이 함수를 호출할 때 전달되는 파라미터
export const getInfo = createAsyncThunk(
  "openAPI/getInfo",
  async (payload, { rejectWidthValue }) => {
    let result = null;

    try {
      result = await axios.get(

`http://apis.data.go.kr/B552584/ArpltnInforInqireSvc/getCtprvnRltmMesureDnsty?
sidoName=서울&pageNo=1&numOfRows=100&returnType=json&serviceKey=${KEY}&ver=1.0`
      );
    } catch (err) {
      // 에러 발생 시 'rejectWidthValue()'함수에 에러 데이터를 전달하면 extraReducer
의 rejected 함수가 호출된다.
      result = rejectWidthValue(err.response);
    }
    return result;
  }
```

```
);

/* Slice 정의 (Action함수 + Reducer의 개념) */
const misaeSlice = createSlice({
  name: "getInfo",
  initialState: {
    items: null,
    loading: false,
    error: null,
  },
  // 내부 action 및 동기 action
  reducers: {},
  // 외부 action 및 비동기 (Ajax용)
  extraReducers: {
    [getInfo.pending]: (state, { payload }) => {
      return { ...state, loading: true };
    },
    [getInfo.fulfilled]: (state, { payload }) => {
      return {
        /* 원하는 key가 깊이 있어서 items까지 접근 */
        items: payload?.data?.response?.body?.items,
        loading: false,
        error: null,
      };
    },
    [getInfo.rejected]: (state, { payload }) => {
      return {
        items: null,
        loading: false,
        error: {
          code: payload?.status ? payload.status : 500,
          message: payload?.statusText ? payload.statusText : "Server Error",
        },
      };
    },
  },
});
// 리듀서 객체 내보내기
export default misaeSlice.reducer;
```

src/store/misaeStore.js

```
/**
 * @filename: misaeStore.js
 * @description: slice 묶어서 index.js에 전달
 * @author: 최수진(sujin971008@gmail.com)
 */
import { configureStore } from "@reduxjs/toolkit";
import misaeSlice from "../slices/misaeSlice";
```

```
const store = configureStore({
  reducer: { getInfo: misaeSlice },
  middleware: (getDefaultMiddleware) =>
    getDefaultMiddleware({ serializableCheck: false }),
  devTools: true,
});

export default store;
```

src/pages/misae.js

```
/**
 * @filename: misae.js
 * @description: 화면에 실질적으로 보여지는 컴포넌트 (액션함수를 dispatch한다)
 * @author: 최수진(sujin971008@gmail.com)
 */
import React from "react";
import styled from "styled-components";
import { useSelector, useDispatch } from "react-redux";

/* slice  */
import { getInfo } from "../slices/misaeSlice";

import Spinner from "../components/Spinner";
import Error from "../components/Error";

/* styledComponent */
const Div = styled.div``;
const SelectContainer = styled.div`
  position: sticky;
  top: 0;
  background-color: #fff;
  padding: 10px 0;
  margin: 0;

  select {
    margin-right: 15px;
    font-size: 16px;
    padding: 5px 10px;
  }
`;

const Table = styled.table``;

const Misae = () => {
  React.useEffect(() => console.clear(), []);
  // hook을 통해 slice가 관리하는 상태값 가져오기
  const { items, loading, error } = useSelector((state) => state.getInfo);
```

```
  /* Grade 함수1 */
  function Grade(x) {
    let state = null;
    if (x === "1") {
      state = "좋음";
    } else if (x === "2") {
      state = "보통";
    } else if (x === "3") {
      state = "나쁨";
    } else if (x === "4") {
      state = "매우나쁨";
    } else {
      state = "오류";
    }
    return state;
  }
  /* Grade값 함수2 */
  function Emoji(y) {
    let state = null;
    if (y === "1") {
      state = "😀";
    } else if (y === "2") {
      state = "😐";
    } else if (y === "3") {
      state = "🥸";
    } else if (y === "4") {
      state = "😂";
    } else {
      state = "오류";
    }
    return state;
  }

  // dispatch 함수 생성
  const dispatch = useDispatch();

  /* 상태값을 '중구'로 기본값을 잡아줌으로써, 페이지가 열리자 마자 정보를 보여주게 된다
(앱과 같은 느낌을 주고 싶었다)  */
  const [stationName, setStationName] = React.useState("중구");

  // 컴포넌트가 마운트되면 데이터 조회를 위한 액션함수를 디스패치 함
  React.useEffect(() => {
    dispatch(getInfo());
  }, [dispatch, stationName]);

  /* 자치구 선택 시 Event */
  const onSelectChange = React.useCallback((e) => {
    e.preventDefault();
    // 드롭다운의 입력값 취득
    const current = e.target;
    const value = current[current.selectedIndex].value;
    setStationName((stationName) => value);
  }, []);
```

```
  return (
    <>
      <Spinner visible={loading} />

      {/* 검색 조건 드롭다운 박스 */}
      <SelectContainer>
        <select
          name="location"
          onChange={onSelectChange}
          style={{ border: "3px solid #168", borderRadius: "7%" }}
        >
          {/* items의 지역명을 반복 돌려서 option 생성 */}
          <option value="">-- 자치구 선택 --</option>
          {items &&
            items.map((v, i) => {
              return (
                <option value={v.stationName} key={i}>
                  {v.stationName}
                </option>
              );
            })}
        </select>
      </SelectContainer>

      {error ? (
        <Error error={error} />
      ) : (
        <Div>
          <div>
            <h1>서울특별시</h1>
            <h3>{stationName}</h3>
          </div>
          {items &&
            items.map((v, i) => {
              return (
                <div>
                  {/* 통합지수 */}
                  <p>
                    {v.stationName === stationName ? Grade(v.khaiGrade) : ""}
                  </p>
                  {/* 통합지수 이모지 */}
                  <span style={{ fontSize: "120px" }}>
                    {v.stationName === stationName ? Emoji(v.khaiGrade) : ""}
                  </span>
                  {/* 미세먼지 */}
                  <div>
                    {v.stationName === stationName ? (
                      <ul>
                        <li>
                          미세먼지: {v.pm10Value}㎍/㎥
                          {Emoji(v.pm10Grade)}
                        </li>
                        <li>
```

```
                        초미세먼지: {v.pm25Value}㎍/㎥
                        {Emoji(v.pm25Grade)}
                      </li>
                    </ul>
                  ) : (
                    <ul></ul>
                  )}
                </div>
              </div>
            );
          })}
      </Div>
    )}
    {/* 기타 대기오염지수 테이블 */}
    {error ? (
      <Error error={error} />
    ) : (
      <Table>
        {items &&
          items.map((v, i) => {
            return v.stationName === stationName ? (
              <thead>
                <tr>
                  <th>아황산가스</th>
                  <td>
                    {Grade(v.so2Grade)}
                    {Emoji(v.so2Grade)}
                  </td>
                  <td>{v.so2Value}</td>
                </tr>
                <tr>
                  <th>일산화탄소</th>
                  <td>
                    {Grade(v.coGrade)}
                    {Emoji(v.coGrade)}
                  </td>
                  <td>{v.coValue}</td>
                </tr>
                <tr>
                  <th>오존</th>
                  <td>
                    {Grade(v.o3Grade)}
                    {Emoji(v.o3Grade)}
                  </td>
                  <td>{v.o3Value}</td>
                </tr>
                <tr>
                  <th>이산화질소</th>
                  <td>
                    {Grade(v.no2Grade)}
                    {Emoji(v.no2Grade)}
                  </td>
                  <td>{v.no2Value}</td>
                </tr>
```

```
                </thead>
              ) : (
                <thead></thead>
              );
            })}
        </Table>
      )}
      <div>
        <i class="fa fa-exclamation-triangle" aria-hidden="true"></i>
        <p>자료 출처: 공공데이터포털을 통한 환경부/한국환경공단 에어코리아 </p>
        <p>
          주의사항: 해당 기관이 제공하는 자료는 “인증을 받지 않은
          실시간자료”이므로 자료 오류 및 표출방식에 따라 값이 다를 수 있습니다.
        </p>
      </div>
    </>
  );
};

export default React.memo(Misae);
```

src/components/Error.js

```
import React from "react";

const Error = ({ error }) => {
  return (
    <div>
      <h1>Oops~!!! {error.code} Error.</h1>
      <p>{error.message}</p>
    </div>
  );
};

export default React.memo(Error);
```

src/components/Spinner.js

```
import React from "react";
import PropTypes from "prop-types";
import styled from "styled-components";

/* 로딩바 컴포넌트 */
// --> https://mhnpd.github.io/react-loader-spinner/
```

```
import { Bars } from "react-loader-spinner";

/* 로딩바 뒤에 표시될 반투명 막 */
const TransLayer = styled.div`
  position: fixed;
  left: 0;
  top: 0;
  z-index: 9999;
  background-color: #0003;
  width: 100%;
  height: 100%;
`;
// visible은 boolean값
const Spinner = ({ visible, color }) => {
  return (
    <div>
      {visible && (
        <TransLayer>
          <Bars
            color={color}
            wrapperStyle={{
              position: "absolute",
              zIndex: 10000,
              left: "50%",
              top: "50%",
              transform: "translate(-50%,-50%)",
            }}
          />
        </TransLayer>
      )}
    </div>
  );
};

/* 기본값 정의 */
Spinner.defaultProps = {
  visible: false,
  color: "#06f",
  width: 100,
  height: 100,
};

/* 데이터 타입 설정 */
Spinner.propTypes = {
  visible: PropTypes.bool.isRequired,
  color: PropTypes.string,
  width: PropTypes.number,
  height: PropTypes.number,
};
// React.memo()를 사용하여 함수형 컴포넌트의 리렌더링 성능을 최적화
export default React.memo(Spinner);
```

CSS를 적용후 변경된 코드입니다

src/Meta.js

```
import React from "react";
import { Helmet, HelmetProvider } from "react-helmet-async";

const Meta = (props) => {
  return (
    <HelmetProvider>
      <Helmet>
        <meta charset="utf-8" />
        <title>{props.title}</title>
        {/* SEO 태그 */}
        <meta name="description" content={props.description} />
        <meta name="keywords" content={props.keywords} />
        <meta name="author" content={props.author} />
        <meta property="og:type" content="website" />
        <meta property="og:title" content={props.title} />
        <meta property="og:description" content={props.description} />
        <meta property="og:image" content={props.image} />
        <meta property="og:url" content={props.url} />
        <link rel="preconnect" href="https://fonts.googleapis.com" />
        <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
        <link
          href="https://fonts.googleapis.com/css2?
family=Gugi&family=Noto+Sans+KR:wght@100;300;400;500&display=swap"
          rel="stylesheet"
          type="text/css"
        />

        {/* Helmet 안에서 CSS적용하기 */}
        <style type="text/css">{`
            *{
                list-style: none;
            }
            body {
                margin: 0;
                padding: 0;
            }

            `}</style>

        {/* 추가적으로 적용해야 할 외부 js나 css로 여기서 명시할 수 있다. */}
        {/*font awesome 참조*/}
        <script
          src="https://kit.fontawesome.com/aa134343d6.js"
          crossorigin="anonymous"
        ></script>
      </Helmet>
    </HelmetProvider>
  );
```

```
};
/* 이게 props */
Meta.defaultProps = {
  title: "useage of OpenAPI",
  description: "OpenAPI 활용 제출물입니다.",
  keywords: "OpenAPI",
  author: "최수진,박세영",
  url: window.location.href,
};

export default Meta;
```

---

src/pages/misea.js

```
/**
 * @filename: misae.js
 * @description: 화면에 실질적으로 보여지는 컴포넌트 (액션함수를 dispatch한다)
 * @author: 최수진(sujin971008@gmail.com),박세영(qkrtpdud9899@gmail.com)
 */
import React from "react";
import styled from "styled-components";
import { useSelector, useDispatch } from "react-redux";

/* slice  */
import { getInfo } from "../slices/misaeSlice";

import Spinner from "../components/Spinner";
import Error from "../components/Error";

/* styledComponent */
const Div = styled.div`
  width: 100%;
  display: flex;
  flex-direction: column;
  align-items: center;
  .area {
    width: 50%;
    display: flex;
    justify-content: space-around;
    border-radius: 3%;
    padding-bottom: 3%;
    h3 {
      padding-top: 20px;
    }
    .dustArea {
      width: 50%;
      text-align: center;
      .integrative {
        p{
```

```
            font-size:30px;
            font-weight:bold;
          }
        }
        .dust {
            li {
              font-size: 20px;
              font-weight: bolder;
            }
        }
        .etc {
          width: 100%;
        }
      }
    }
    .textArea {
      margin-top: 4%;
      text-align: center;
      opacity: 0.6;
      background-color: lightgray;
      color: black;
      p {
        font-size: 12px;
      }
      div {
        font-size: 13px;
        i {
          color: red;
        }
        span {
          font-weight: bold;
        }
      }
    }
  `;
  const SelectContainer = styled.div`
    position: sticky;
    top: 0;
    margin: 0;
    display: flex;
    align-items: flex-start;
    padding-top: 10%;
    select {
      border-radius: 8px;
      margin-right: 15px;
      font-size: 16px;
      padding: 5px 5px;
      border: 3px solid black;
      option {
        background-color: black;
        color: #fff;
      }
    }
  `;
```

```
const Misae = () => {
  React.useEffect(() => console.clear(), []);
  // hook을 통해 slice가 관리하는 상태값 가져오기
  const { items, loading, error } = useSelector((state) => state.getInfo);

  /* Grade 함수1 */
  function Grade(x) {
    let state = null;
    if (x === "1") {
      state = "좋음";
    } else if (x === "2") {
      state = "보통";
    } else if (x === "3") {
      state = "나쁨";
    } else if (x === "4") {
      state = "매우나쁨";
    } else {
      state = "오류";
    }
    return state;
  }

  /* Grade값 함수2 */
  function Emoji(y) {
    let state = null;
    if (y === "1") {
      state = "😀";
    } else if (y === "2") {
      state = "😐";
    } else if (y === "3") {
      state = "😨";
    } else if (y === "4") {
      state = "🤮";
    } else {
      state = "오류";
    }
    return state;
  }

  // dispatch 함수 생성
  const dispatch = useDispatch();

  /* 상태값을 '중구'로 기본값을 잡아줌으로써, 페이지가 열리자 마자 정보를 보여주게 된다
(앱과 같은 느낌을 주고 싶었다)  */
  const [stationName, setStationName] = React.useState("중구");

  // 컴포넌트가 마운트되면 데이터 조회를 위한 액션함수를 디스패치 함
  React.useEffect(() => {
    dispatch(getInfo());
  }, [dispatch, stationName]);

  /* 자치구 선택 시 Event */
  const onSelectChange = React.useCallback((e) => {
```

```
    e.preventDefault();
    // 드롭다운의 입력값 취득
    const current = e.target;
    const value = current[current.selectedIndex].value;
    setStationName((stationName) => value);
  }, []);

  //배경색을 바꾸기위해 참조
  const bgRef = React.useRef();

  return (
    <Div>
      <Spinner visible={loading} />
      <h1>서울특별시 대기질 현황</h1>
      {error ? (
        <Error error={error} />
      ) : (
        <div className="area" ref={bgRef}>
          {/* 검색 조건 드롭다운 박스 */}
          <SelectContainer>
            <select name="location" onChange={onSelectChange}>
              {/* items의 지역명을 반복 돌려서 option 생성 */}
              <option value="">-- 자치구 선택 --</option>
              {items &&
                items.map((v, i) => {
                  return (
                    <option value={v.stationName} key={i}>
                      {v.stationName}
                    </option>
                  );
                })}
            </select>
          </SelectContainer>
          {items &&
            items.map((v, i) => {
              //v.khaiGrade 값에 따른 배경색 분기
              if (v.khaiGrade === "1") {
                bgRef.current.style.backgroundColor = "#01DF01";
              } else if (v.khaiGrade === "2") {
                bgRef.current.style.backgroundColor = "#F7D358";
              } else if (v.khaiGrade === "3") {
                bgRef.current.style.backgroundColor = "#FD9903";
              } else if (v.khaiGrade === "4") {
                bgRef.current.style.backgroundColor = "#FF1C11";
              } else {
                bgRef.current.style.backgroundColor = "#fff";
              }
              return (
                v.stationName === stationName && (
                  <div className="dustArea">
                    <h3>{v.stationName}</h3>
                    <div className="integrative">
                      {/* 통합지수 */}
                      <p>
```

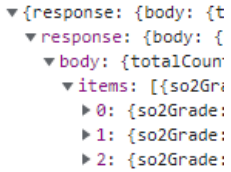
```
                {v.stationName === stationName
                  ? Grade(v.khaiGrade)
                  : ""}
              </p>
              {/* 통합지수 이모지 */}
              <span style={{ fontSize: "120px" }}>
                {v.stationName === stationName
                  ? Emoji(v.khaiGrade)
                  : ""}
              </span>
              {/* 미세먼지 */}
              <div className="dust">
                {v.stationName === stationName ? (
                  <ul>
                    <li>
                      미세먼지: {v.pm10Value}㎍/㎥
                      {Emoji(v.pm10Grade)}
                    </li>
                    <li>
                      초미세먼지: {v.pm25Value}㎍/㎥
                      {Emoji(v.pm25Grade)}
                    </li>
                  </ul>
                ) : (
                  <ul></ul>
                )}
              </div>
            </div>
            {/* 기타 대기오염지수 테이블 */}
            <table className="etc">
              {v.stationName === stationName ? (
                <thead>
                  <tr>
                    <th>아황산가스</th>
                    <td>
                      {Grade(v.so2Grade)}
                      {Emoji(v.so2Grade)}
                    </td>
                    <td>{v.so2Value}ppm</td>
                  </tr>
                  <tr>
                    <th>일산화탄소</th>
                    <td>
                      {Grade(v.coGrade)}
                      {Emoji(v.coGrade)}
                    </td>
                    <td>{v.coValue}ppm</td>
                  </tr>
                  <tr>
                    <th>오존</th>
                    <td>
                      {Grade(v.o3Grade)}
                      {Emoji(v.o3Grade)}
                    </td>
```

```
                        <td>{v.o3Value}ppm</td>
                      </tr>
                      <tr>
                        <th>이산화질소</th>
                        <td>
                          {Grade(v.no2Grade)}
                          {Emoji(v.no2Grade)}
                        </td>
                        <td>{v.no2Value}ppm</td>
                      </tr>
                    </thead>
                  ) : (
                    <thead></thead>
                  )}
                </table>
              </div>
            )
          );
        })}
      </div>
    )}
    <div className="textArea">
      <div>
        <i
          className="fa-solid fa-triangle-exclamation"
          aria-hidden="true"
        ></i>
        <span>주의사항</span>
        <p>
          해당 기관이 제공하는 자료는 “인증을 받지 않은 실시간자료”이므로 자료
          오류 및 표출방식에 따라 값이 다를 수 있습니다.
        </p>
      </div>
      <p>자료 출처: 공공데이터포털을 통한 환경부/한국환경공단 에어코리아 </p>
    </div>
  </Div>
 );
};

export default React.memo(Misae);
```

최 수 진 소 감

- redux가 아직 익숙지 않아서 좀 더 발전시키고 싶은 마음에 redux로 구현해 보았습니다.
- 처음에는 위치에 따른 지역 대기질을 구현하려 했지만, 두 개의 api를 사용해야 하는 부분이 두 명에서 하기에는 시간이 걸리겠다는 생각이 들었습니다. 따라서 서울이라는 한정적인 지역으로 추려서 구현하게 되었습니다.
- 액션함수를 디스패치할 때, 필요한 데이터만 추려서 상태값에 반영하고 싶었습니다. 하지만 받아온 payload 데이터 중에 필요한 items는 몇 겹 내부에 존재했기 때문인지 가져온 데이터를 쓸 수는 있어도 다시 추려서 받아오는 계획은 실패했습니다.



- 차선책으로 삼항연산자와 함수로 구현하였고 이 부분이 처음에 계획과는 달라져서 아쉽습니다.
- 대기 지수에 따라서 이모지를 표현하였는데 등급은 4단계였지만 출력된 값은 1,2단계뿐이어서 살짝 아쉽기도 하고 서울공기가 이렇게 좋았나? 싶었습니다.
- 클론코딩과는 달리 원하는 바를 상상하면서 로직을 짜는 과정이 재밌게 느껴졌습니다.
- 소스코드를 팀원과 공유하면서 코딩한 적이 처음이었는데 장단점이 있다고 생각되었습니다.
- 장점은 고민되는 부분을 의논할 수 있다는 점과 작성 코드가 줄어든다는 점이었습니다.
- 단점은 git사용이 아직 서툴러서 pull push할 때 반드시 상대방에게 얘기를 해야 하는 점이 번거로웠습니다.

박 세 영 소 감

- 처음 어떤 방향으로 나아갈지는 수월하게 논의되고 결정이 되었습니다. 서로 응원하고 다독이며 했고 완성했기에 많이 뿌듯합니다.
- 어떻게 각자의 영역을 나누고 시작할지 많이 고민을 했습니다. 아직 코드를 짜는 것이 서툰 저를 위해 수진님께서 코드 뼈대를 작성해주시고 제가 그 위의 스타일을 입히는 방식으로 했는데 수진님의 코드를 보며 좋은 공부가 되었습니다.
- 수진님의 코드 위에 제가 CSS를 덧붙이며 대기질의 상태에 따라 배경색을 바꾸는 것을 하고싶었고 최종적으로 useRef를 이용하였습니다.
  useRef가 바로 생각이 나지않아 많은 시간이 걸렸고 useRef를 적용하고 코드를 작성할때 수진님께서 작성해주신 Grader함수들을 이용해 적용하고 싶었지만 실패했고 제가 따로 코드를 작성하여 추가해 대기질의 상태에 따라 배경색을 바꾸도록 구현했습니다.
- 대기질의 상태에 따라 배경색을 바꾸도록 구현했지만 서울의 대기질이 전부 보통으로 나와 구현 결과 스크린샷이 전부 보통일때 뿐인게 아쉽습니다..
- 수진님을 보며 저의 문제점을 발견했습니다. 같이 하는 팀원이 있기에 발견할수 있었던 문제점이였고 문제점을 개선할 기회를 얻었다는것이 매우 좋다고 생각합니다.