WEB APPLICATION FINAL PROJECT REPORT
AIR QULITY MONITORING SITE

202125203 Sejeong PARK

PROJECT OVERVIEW

This project is a web-based air quality monitoring dashboard that allows users to track real-time air quality and weather data for specific cities and districts. By using public APIs, the platform fetches data such as PM2.5, PM10, NO_2, SO_2, O_3, CO, temperature, humidity, wind speed, and more, visualizing the information through charts and intuitive indicators. The goal is to help users monitor their surroundings in a clear, accessible, and meaningful way.

KIND OF SOFTWARE TOOLS AND TECHNOLOGIES USED

- **Frontend**
  - **HTML5** : Structuring the webpage and UI components.
  - **CSS** : Styling and designing the layout with responsive features.
  - **JavaScript** : Implementing interactivity, API integration, and data visualization.

- **Libraries**
  - **Chart.js** : Used to create visual representations such as line charts, bar charts, and doughnut charts.

  - **APIs**
    1. **Korea Meteorological Administration API** (기상청): Provides real-time weather data such as temperature, humidity, and wind speed.
    2. **AirKorea API** (한국환경공단): Delivers real-time air quality information such as PM2.5, PM10, and gas concentrations.

DATA SOURCES

- Weather Data

Sourced from the **Korea Meteorological Administration** through their **지상(종관, ASOS) 시간자료 조회서비스 API**.

기본정보

데이터명	기상청_지상(종관, ASOS) 시간자료 조회서비스 상세설명		
서비스유형	REST	심의여부	자동승인
신청유형	개발계정 | 활용신청	처리상태	승인
활용기간	2024-11-17 ~ 2026-11-17		

서비스정보

참고문서	기상청01_지상(종관,ASOS)시간자료 조회서비스 오픈API활용가이드.docx
데이터포맷	JSON+XML
End Point	http://apis.data.go.kr/1360000/AsosHourlyInfoService

요청변수(Request Parameter)

닫기

항목명	샘플데이터	설명
ServiceKey	-	공공데이터포털에서 받은 인증키
pageNo	1	페이지번호 Default : 10
numOfRows	10	한 페이지 결과 수 Default : 1
dataType	JSON	요청자료형식(XML/JSON) Default : XML
dataCd	ASOS	자료 분류 코드(ASOS)
dateCd	HR	날짜 분류 코드(HR)
startDt	20100101	조회 기간 시작일(YYYYMMDD)
startHh	01	조회 기간 시작시(HH)
endDt	20100601	조회 기간 종료일(YYYYMMDD) (전일(D-1) 까지 제공)
endHh	01	조회 기간 종료시(HH)
stnIds	108	종관기상관측 지점 번호 (활용가이드 하단 첨부 참조)

샘플코드

Java | Javascript | C# | PHP | Curl | Objective-C | Python | Nodejs | R

```
/* Javascript 샘플 코드 */


var xhr = new XMLHttpRequest();
var url = 'http://apis.data.go.kr/1360000/AsosHourlyInfoService/getWthrDataList'; /*URL*/
var queryParams = '?' + encodeURIComponent('serviceKey') + '='+'서비스키'; /*Service Key*/
queryParams += '&' + encodeURIComponent('pageNo') + '=' + encodeURIComponent('1'); /**/
queryParams += '&' + encodeURIComponent('numOfRows') + '=' + encodeURIComponent('10'); /**/
queryParams += '&' + encodeURIComponent('dataType') + '=' + encodeURIComponent('XML'); /**/
queryParams += '&' + encodeURIComponent('dataCd') + '=' + encodeURIComponent('ASOS'); /**/
queryParams += '&' + encodeURIComponent('dateCd') + '=' + encodeURIComponent('HR'); /**/
queryParams += '&' + encodeURIComponent('startDt') + '=' + encodeURIComponent('20100101'); /**/
queryParams += '&' + encodeURIComponent('startHh') + '=' + encodeURIComponent('01'); /**/
queryParams += '&' + encodeURIComponent('endDt') + '=' + encodeURIComponent('20100601'); /**/
queryParams += '&' + encodeURIComponent('endHh') + '=' + encodeURIComponent('01'); /**/
queryParams += '&' + encodeURIComponent('stnIds') + '=' + encodeURIComponent('108'); /**/
xhr.open('GET', url + queryParams);
xhr.onreadystatechange = function () {
    if (this.readyState == 4) {
        alert('Status: '+this.status+'nHeaders: '+JSON.stringify(this.getAllResponseHeaders())+'nBody: '+this.responseText);
    }
};

xhr.send('');
```

The data was obtained using an open API in JSON format. By referring to the sample code in JavaScript and including the required request parameters in the URL, regional weather data can be retrieved.

- Air Quality Data

Retrieved from **AirKorea** (한국환경공단 대기오염정보 API). This includes PM2.5, PM10, and gaseous pollutants (NO_2, SO_2, O_3, CO).

기본정보

데이터명	한국환경공단_에어코리아_대기오염정보 상세설명		
서비스유형	REST	심의여부	자동승인
신청유형	개발계정 | 활용신청	처리상태	승인
활용기간	2024-11-17 ~ 2026-11-17		

서비스정보

참고문서	한국환경공단 에어코리아 OpenAPI 기술문서.zip
데이터포맷	JSON+XML
End Point	http://apis.data.go.kr/B552584/ArpltnInforInqireSvc

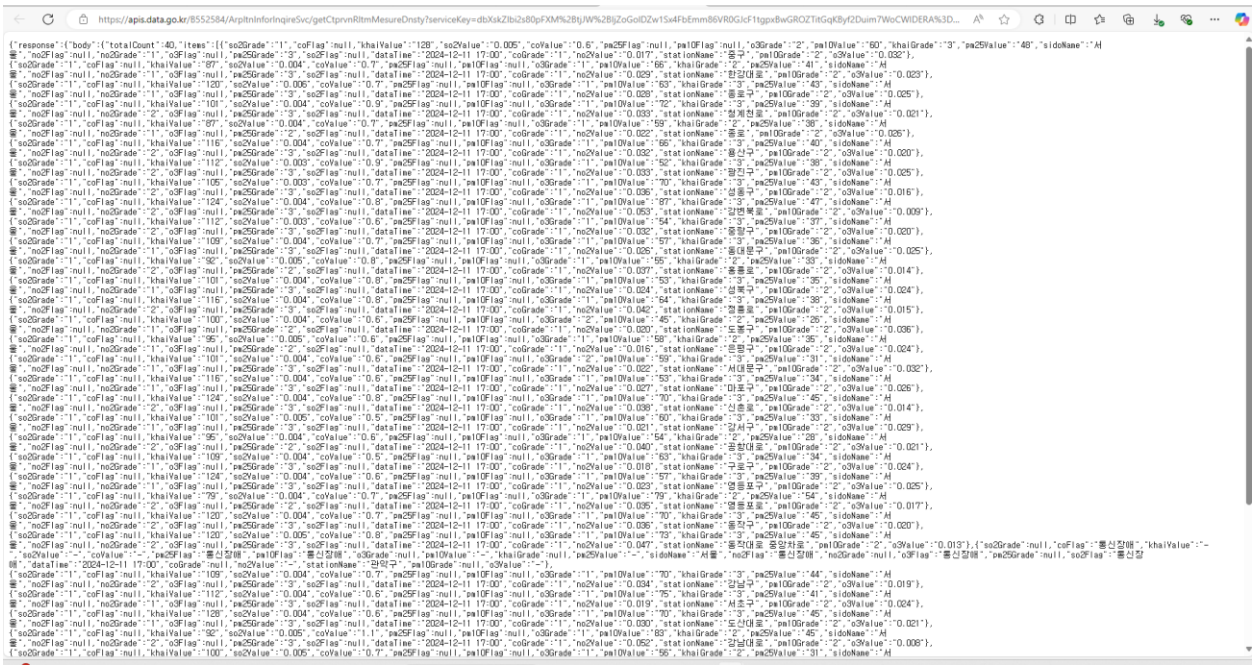
활용신청 상세기능정보

NO	상세기능	설명	일일 트래픽	미리보기
1	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

요청변수(Request Parameter)

닫기

항목명	샘플데이터	설명
serviceKey	-	공공데이터포털에서 받은 인증키
returnType	json	xml 또는 json
numOfRows	100	한 페이지 결과 수
pageNo	1	페이지번호
sidoName	서울	시도 이름(전국, 서울, 부산, 대구, 인천, 광주, 대전, 울산, 경기, 강원, 충북, 충남, 전북, 전남, 경북, 경남, 제주, 세종)
ver	1.0	버전별 상세 결과 참고



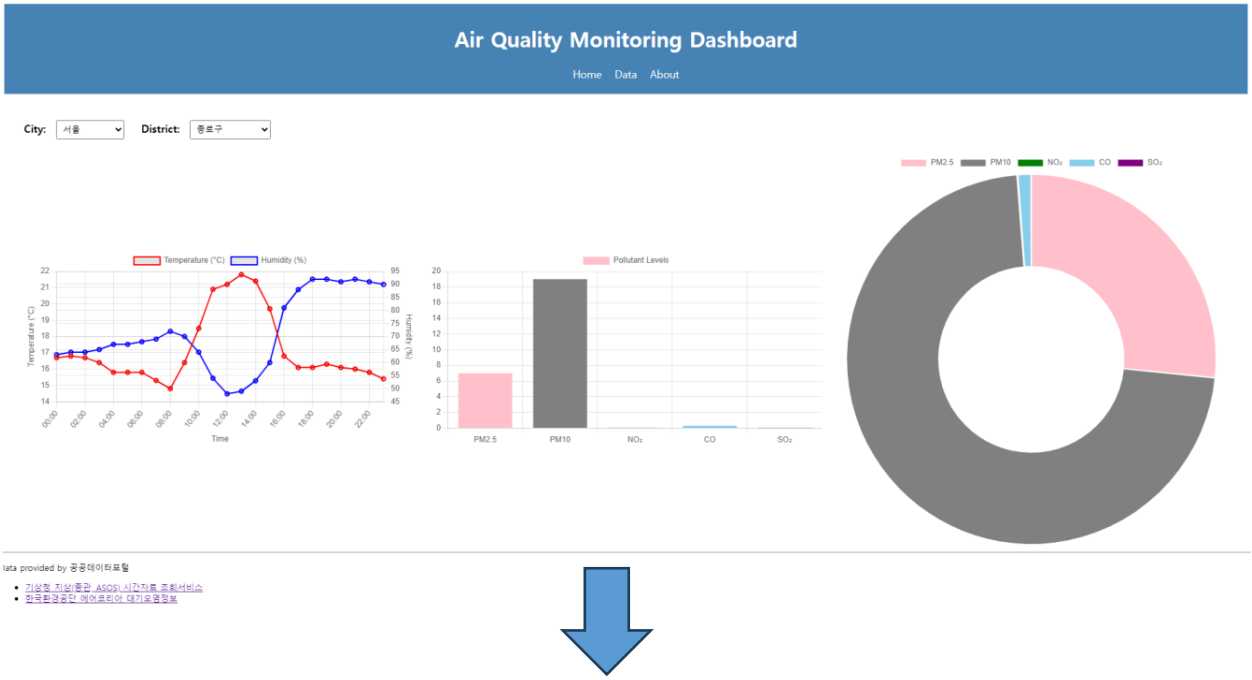
샘플코드

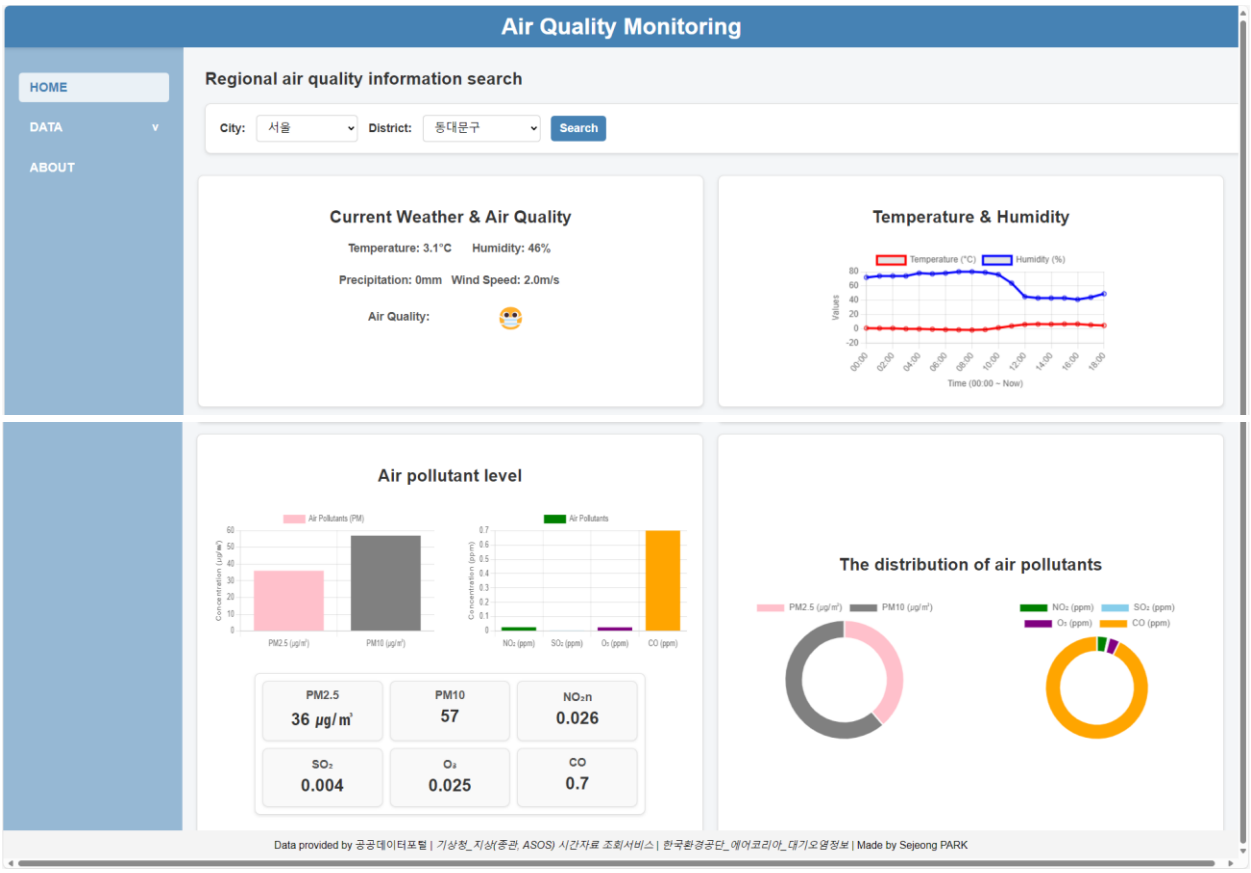
Java | Javascript | C# | PHP | Curl | Objective-C | Python | Nodejs | R

```
/* Javascript 샘플 코드 */


var xhr = new XMLHttpRequest();
var url = 'http://apis.data.go.kr/B552584/ArpltnInforInqireSvc/getMinuDustFrcstDspth'; /*URL*/
var queryParams = '?' + encodeURIComponent('serviceKey') + '='+'서비스키'; /*Service Key*/
queryParams += '&' + encodeURIComponent('returnType') + '=' + encodeURIComponent('xml'); /**/
queryParams += '&' + encodeURIComponent('numOfRows') + '=' + encodeURIComponent('100'); /**/
queryParams += '&' + encodeURIComponent('pageNo') + '=' + encodeURIComponent('1'); /**/
queryParams += '&' + encodeURIComponent('searchDate') + '=' + encodeURIComponent('2020-11-14'); /**/
queryParams += '&' + encodeURIComponent('InformCode') + '=' + encodeURIComponent('PM10'); /**/
xhr.open('GET', url + queryParams);
xhr.onreadystatechange = function () {
    if (this.readyState == 4) {
        alert('Status: '+this.status+'nHeaders: '+JSON.stringify(this.getAllResponseHeaders())+'nBody: '+this.responseText);
    }
};

xhr.send('');
```

The air quality data was also obtained using an open API in JSON format. By referring to the sample code in JavaScript and including the required request parameters in the URL, regional air quality data can be retrieved.

RESOLVING IMPLEMENTATION ISSUES AND IMPROVEMENTS





1. Resolving Current Problems

1.1 CORS (Cross-Origin Resource Sharing) Error

- A CORS error occurred due to a mismatch between HTTP and HTTPS protocols when attempting to fetch data.
- Solution: Unified the URL protocol to HTTP for all API requests to avoid cross-origin issues. Debugging was done using console.log() to identify and resolve the exact point where the issue occurred.

1.2 Real-Time Data Fetching

- Initially, the weather data fetched was static and did not reflect real-time conditions.
- Solution: Modified the code to include variables for dynamic API calls. These variables allow users to fetch real-time weather and air quality data for the selected regions.

2. Creation of Home, Data, and About Pages

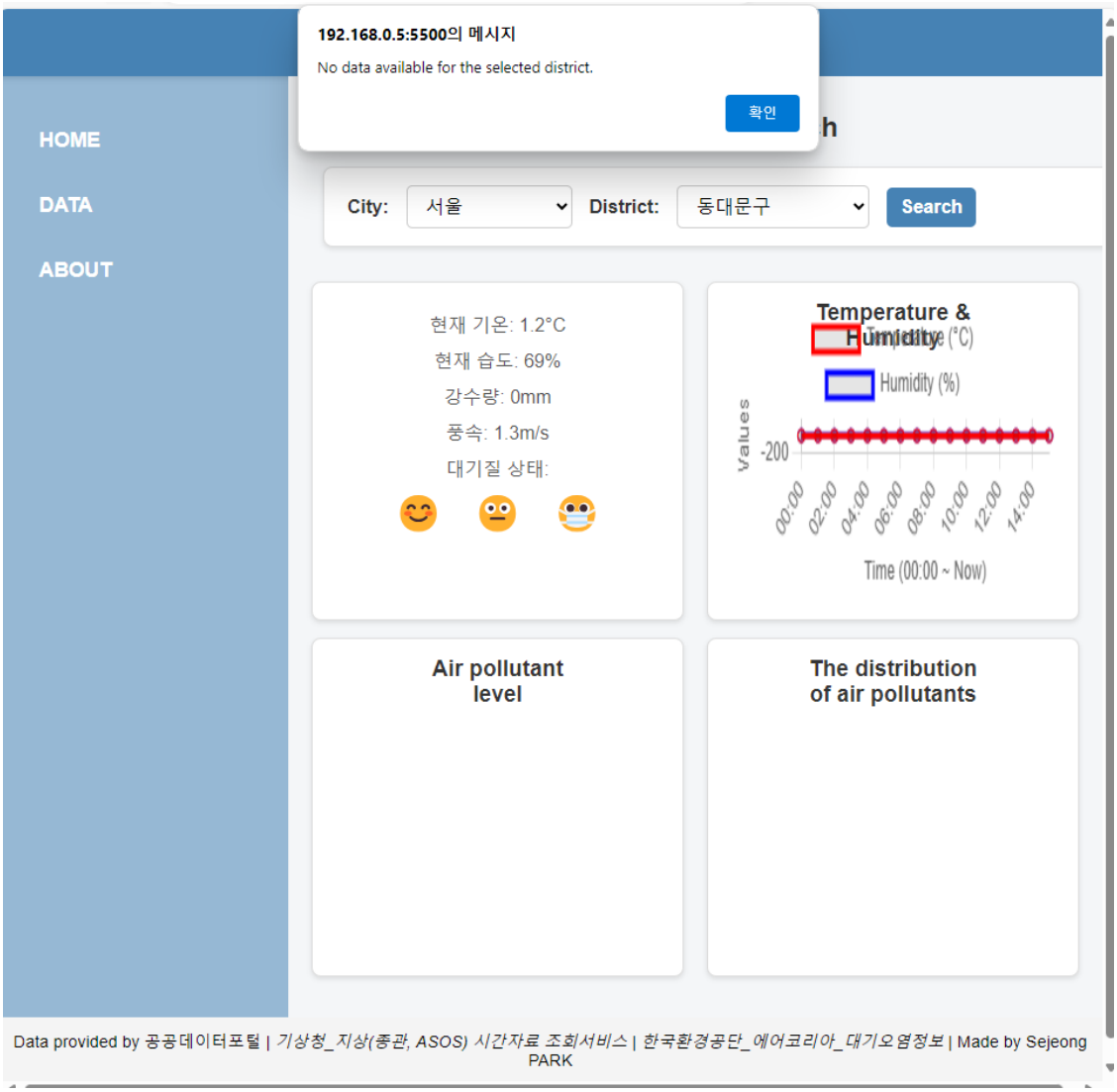
- Clicking on the buttons did not trigger any response.
- Solution: Implemented JavaScript to handle navigation between pages dynamically using hyperlinks. Ensured each page displayed its respective content correctly.

3. Overall Design Modifications Using CSS

- Initial Design Issues : The initial version of the website had a very basic layout with minimal styling. The design lacked visual appeal and did not resemble a professional, user-friendly site.
- Design Improvements

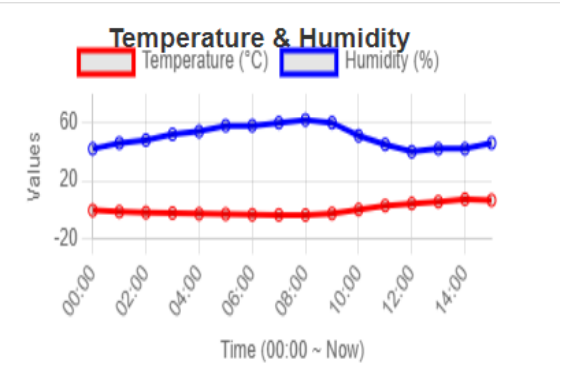
1. Side Menu Addition: A side navigation menu was added, allowing users to select Home, Data, and About pages seamlessly.
2. Layout Redesign: Improved the overall layout to display real-time weather and air quality data more effectively. Utilized CSS for cleaner styling, proper spacing, and alignment to enhance user experience.
3. Visual Clarity: Applied consistent styles, fonts, and color schemes for a more professional look and feel.

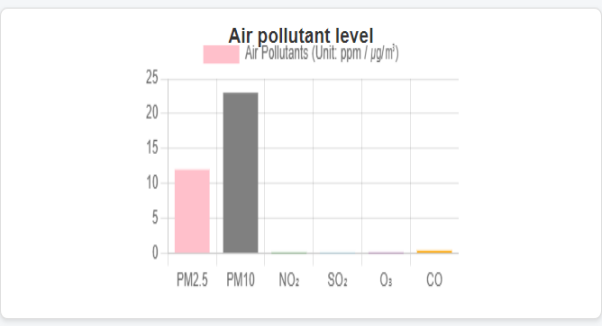
FAILURE AND SUCCESS

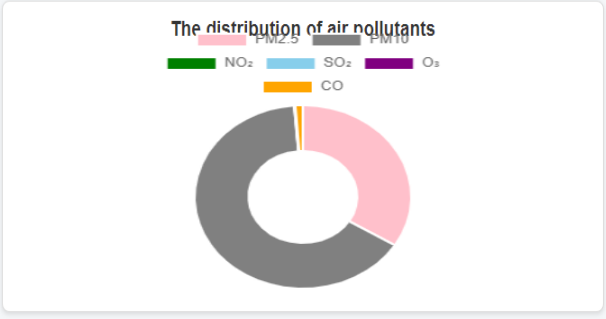


The most significant challenge I faced was retrieving the data. Extracting only the necessary information from the data source and processing it was particularly difficult. To resolve this issue, I spent nearly one to two weeks making numerous attempts. The issue shown in the screenshot was resolved by writing the code as follows. To resolve this, I used the .filter() function to compare stationName against the items in the API response.

```
function fetchAirPollutantData(sidoName, stationName) {
    const url = `https://apis.data.go.kr/B552584/ArpltnInforInqireSvc/getCtprvnRltmMesureDnsty`;
    const serviceKey = `dbXskZIbi2s80pFXM%2BtjJW%2BIjZoGolDZw1Sx4FbEmm86VR0GJcF1tgpxBwGROZTitGqKByf2Duim7WoCWlDERA%3D%3D`;
    const queryParams = `?serviceKey=${serviceKey}&returnType=json&numOfRows=100&pageNo=1&sidoName=${encodeURIComponent(sidoName)}&ver=1.0`;

    const requestUrl = url + queryParams;

    fetch(requestUrl)
        .then(response => response.json())
        .then(data => {
            if (data.response && data.response.body && data.response.body.items) {
                const filteredData = data.response.body.items.filter(item => item.stationName === stationName);

                if (filteredData.length > 0) {
                    updateAirPollutantCharts(filteredData[0]);
                }
            }
        })

}
```







I also faced significant challenges with the CSS of the charts. The chart titles and the charts themselves kept overlapping, and when I structured the areas using various tags, it took time to identify which tags were being affected by the padding and margin values. The areas were divided using chart-box and info-box classes, and CSS was applied accordingly. By utilizing the corresponding IDs or class values within these sections, the appropriate code was able to function correctly in the designated areas.

```css
.chart-box, .info-box {
    background-color: white;
    border: 1px solid #ddd;
    border-radius: 8px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
    padding: 20px;
    display: flex;
    flex-direction: column;
    justify-content: flex-start;
    align-items: center;
    text-align: center;
    position: relative;
}

.chart-box h3 {
    margin-bottom: 20px;
    font-size: 1.5rem;
    color: #333;
    font-weight: bold;
    text-align: center;
}

.chart-box {
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    gap: 10px;
    padding: 20px 10px;
}
```

```css
#chart {
    display: flex;
    justify-content: space-around;
    align-items: center;
    width: 100%;
    gap: 20px;
}

#chart canvas {
    flex: 1;
    max-width: 45%;
    height: auto;
    box-sizing: border-box;
}

.chart-box canvas {
    max-width: 90%;
    height: 200px !important;
    margin: 0 auto;
    box-sizing: border-box;
}

.info-chart-container {
    display: grid;
    grid-template-columns: 1fr 1fr;
    grid-gap: 20px;
    grid-column: span 2;
}

.chart-container {
    display: grid;
    grid-template-columns: 1fr 1fr;
    grid-gap: 20px;
    grid-column: span 2;
}
```

PROJECT SIGNIFICANCE

1. **Real-time Monitoring**: Users can view up-to-date air quality and weather information specific to their city or district.
2. **Data Visualization**: Complex data is visualized using user-friendly charts, making it easy to interpret.
3. **Awareness**: Helps raise awareness of environmental conditions, encouraging informed decisions regarding outdoor activities.
4. **Accessibility**: Web-based platform ensures availability across devices (PC, mobile, and tablet).