# WEB APPLICATION PROGRAMMING (CB2001105-062) Project Report

Dawon Jeong (201924569)
Pusan National University

## 1 Introduction

The Air Quality Monitoring Dashboard is a web-based application developed to visualize real-time environmental data such as temperature, humidity, and pollutant levels. The project employs modern web development techniques, including Single Page Application (SPA) architecture, interactive data visualization, and responsive design principles, to deliver a seamless user experience. To address the lack of access to real-world data for all metrics, the application integrates API-based data fetching and supplements missing data with randomized values to simulate a comprehensive monitoring experience. The project aims to educate users about air quality dynamics through real-time insights and interactive visualizations.

## 2 System Architecture

### 2.1 Frontend Development

The frontend is constructed using HTML5, CSS3, and JavaScript, with additional libraries like Bootstrap for layout responsiveness and Chart.js for interactive charts. The main components include:

- **Sidebar Navigation:** A persistent menu for easy navigation between "Home," "Data," and "About" sections.
- **Data Cards:** Nine distinct cards presenting air quality metrics such as temperature, PM2.5, and CO2 levels in a visually appealing format.
- **Charts Section:** Includes line charts for temporal trends, bar charts for pollutant comparisons, and doughnut charts for pollutant proportions.
- **Footer:** Provides external resource links for further exploration and information.

### 2.2 Backend Functionality

The dynamic functionalities are implemented using JavaScript, which handles the following core operations:

- **Real-Time Data Simulation:** Generates randomized data every 5 seconds for parameters such as temperature, humidity, and pollutant levels.
- **Chart Updates:** Dynamically refreshes chart data based on the simulated values.
- **Single Page Application (SPA):** Implements a navigation system that dynamically loads content without requiring full-page reloads.
- **API Integration:** Utilizes an air quality API to fetch geolocation-specific data where available, ensuring accuracy for supported metrics.

### 2.3 Styling

The design prioritizes user engagement through a clean and modern interface. Key styling features include:

- **Purple Gradient Theme:** Applied to the sidebar and data cards for visual consistency.
- **Flexbox Layouts:** Ensures responsive alignment of components across various devices.
- **Custom Fonts and Colors:** Enhances readability and accessibility.

## 3 Features

- **Real-Time Visualization:** Continuously updates air quality data and pollutant metrics.
- **Interactive Charts:** Utilizes Chart.js to create visually appealing and interactive data visualizations.
- **Responsive Design:** Guarantees compatibility across desktops, tablets, and smartphones.
- **API Data Integration:** Fetches live air quality metrics using an API, supplemented by simulated data for unsupported parameters.

## 4 Project Requirements Satisfaction

### 4.1 User Interface Design

- Provides a user-friendly layout with a sidebar, dashboard, and footer sections.
- Ensures responsive behavior for seamless interaction on devices of all sizes.

### 4.2 Data Display

- Displays key metrics such as Temperature, PM2.5, and CO2 in easily interpretable cards.

### 4.3 Visual Representation

- Implements interactive charts for trends and comparisons using Chart.js.

### 4.4 Real-Time Data Updates

- Fetches live data using APIs for available parameters.
- Simulates realistic random values for metrics not supported by the API to maintain completeness.

### 4.5 Styling and Accessibility

- Employs color-coded indicators to highlight air quality levels (e.g., green for safe, red for hazardous).
- Ensures compliance with accessibility standards through semantic HTML and ARIA labels.

## 5 Key Code Snippets

### 5.1 HTML: Dashboard Structure

```
<section id="data-cards">
    <div class="card">
        <h3>Temperature</h3>
        <p><span id="temperature">25</span> °C</p>
    </div>
    <div class="card">
        <h3>PM2.5</h3>
        <p><span id="pm25">35</span> µg/m³</p>
    </div>
</section>
```

### 5.2 JavaScript: Real-Time Data Simulation and API Integration

```
setInterval(() => {
    const randomData = {
        temperature: (Math.random() * 30 + 10).toFixed(2),
        pm25: (Math.random() * 50).toFixed(2),
        co2: (Math.random() * 500 + 300).toFixed(2)
    };
    updateCharts(randomData);
}, 5000);

fetch(`https://api.example.com/data?lat=35.1796&lon=129.0756&key=${API_KEY}`)
    .then(response => response.json())
    .then(apiData => {
        console.log(apiData); // Logs fetched data
    })
    .catch(error => {
        console.error('API Error:', error);
    });
```

### 5.3 CSS: Styling

```
.card {
    width: 170px;
    padding: 15px;
    background: #836fff;
    color: #fff;
    text-align: center;
    border-radius: 10px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}
```

## 6 Challenges and Solutions

- **Challenge: Simulating Real-Time Data**
  - **Solution:** Developed randomized data generation for metrics unavailable through the API, ensuring realistic behavior and variability.
- **Challenge: Chart Responsiveness**
  - **Solution:** Leveraged Chart.js properties to dynamically resize charts based on viewport dimensions.
- **Challenge: Combining API and Simulated Data**
  - **Solution:** Integrated a fallback mechanism where missing API data is supplemented by random values.

## 7 Future Improvements

- Expand API integration to include more diverse air quality metrics.
- Add advanced analytics, such as historical trends and predictive modeling.
- Introduce user customization options, including alert thresholds and preferred metrics.

## 8 Conclusion

The Air Quality Monitoring Dashboard demonstrates the integration of modern web development tools and practices to create an engaging and functional SPA. By combining API-based data fetching with randomized simulations, the application delivers a comprehensive visualization experience. Future enhancements will focus on real-world data expansion and advanced features to further improve utility and user engagement.