horizontal line

**Takibay Sultan**

202155633

Web Application Programming (CB2001105-062)

AirSapa

**10th December 2024**

# OVERVIEW

The AirSapa Web Application is designed to provide users with real-time air quality data for selected locations. The application offers an intuitive user interface for retrieving and visualizing air quality metrics through an interactive map and API integrations. The main goal of this project is to enhance awareness and understanding of air quality, promoting healthier lifestyles.

# Tools and Software

1. **Backend**: Python, Django
2. **Frontend Technologies**: HTML, CSS (Bootstrap 5), JavaScript, Google Icons
3. **Database**: SQLite (development) and PostgreSQL (production)
4. **Hosting Platform**: PythonAnywhere
5. **Version Control**: Git and GitHub for repository management

# API

1. **WAQI**: For Air Quality Dashboard
2. **WeatherAPI**: For Weather Forecast and air quality
3. **OpenCage**: For Coordinates

# Other

**Leaflet.js**: For the map

**OpenStreetMap**: For the map

**Redis**: Redis was used as a message broker for Celery tasks. Its in-memory architecture ensures rapid communication between the Django application and Celery workers. Redis is also leveraged for caching data and storing task-related information temporarily. Unfortunately, in PythonAnywhere redis is not available

**Celery**: Celery facilitated the execution of long-running tasks in the background, ensuring that the main application remains responsive. Just like Redis, Celery was also not available in PythonAnywhere

# Features Implemented

1. **Interactive Map**: Users can select a location on a map to retrieve air quality data dynamically.
2. **Responsive Design**: The web app supports multiple devices, including desktop, tablet, and mobile views.
3. **API Integration**: Fetches air quality indices and displays them in a user-friendly format.
4. **Error Handling**: Ensures that users are informed when the API fails or returns invalid data.
5. **User Authentication**: Supports login, registration, and secure session management.
6. **Email Sending**: Users can choose to receive emails about weather/aqi/health advice alerts

# Successes

1. **API Integration**: APIs for air quality, weather, and location were successfully integrated, providing real-time data.
2. **Deployment**: The application was deployed on PythonAnywhere, but with some initial configuration issues.
3. **Performance**: Optimized API calls and database queries to reduce latency.
4. **Responsive Design**: The design adjusts to various screen sizes, ensuring functionality across devices.

# Challenges and Failures

1. **Deployment Issues**: Faced issues with configuring the virtual environment during deployment on PythonAnywhere, which were resolved by properly linking the WSGI file and installing all required dependencies. As well as frustrated about PythonAnywhere that its free version doesn’t support Redis and Celery
2. **Interactive Map**: Integrating the map with dynamic user input required troubleshooting multiple JavaScript errors.
3. **Error Handling**: Initial lack of error messages for API downtime led to confusion; this was addressed with user-friendly error messages.
4. **Historical AQI Data**: I couldn’t find the available API with sufficient database to support this feature. I have made the template and view function ready, but eventually failed because of insufficient data from the API.
5. **Lack of pages** such as About, FAQ, Documentation, etc.: Because of limited time I wasn’t able to fully finish my project.

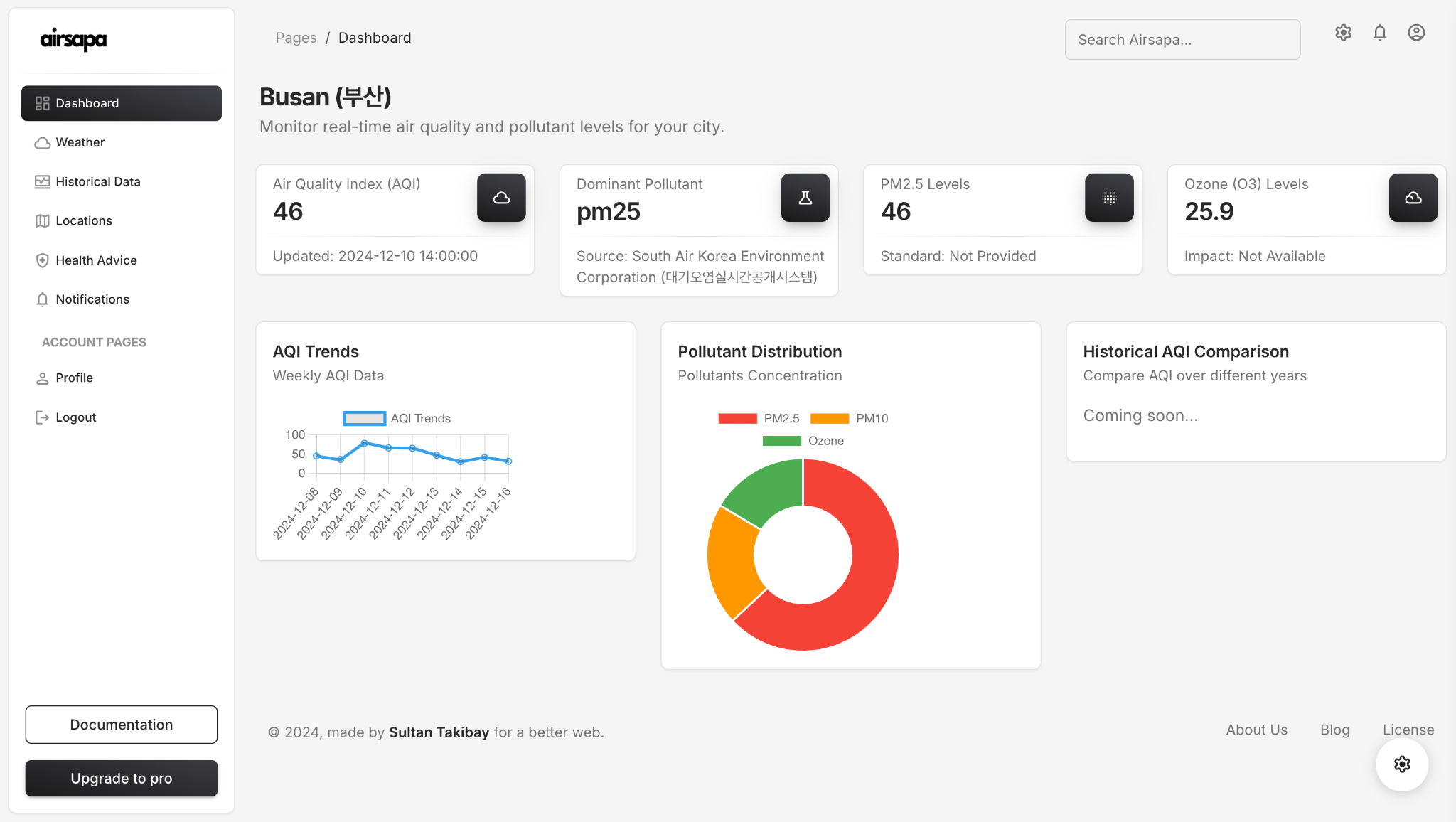
# HOW TO USE?

1. Clone the repository: git clone <https://github.com/sultannurzhan/airsapa.git>
2. Activate the virtual environment (uploaded in github alongside): source venv/bin/activate
3. Run the application locally: python3 manage.py runserver
4. Access it at: <http://localhost:8000/>

## Website URL

<https://airsapa.pythonanywhere.com>

## Screenshots



## 

## Conclusion

The AirSapa Web Application is a straightforward tool designed to provide and display real-time air quality data. Throughout the development process, the project faced several challenges, including API integration issues, deployment complexities, and design constraints. However, these challenges were successfully overcome, resulting in a functional application that seamlessly integrates air quality APIs, employs efficient deployment strategies, and features a responsive design optimized for various devices.

Future enhancements could focus on incorporating advanced data visualization techniques, such as interactive charts, predictive analytics, and heatmaps for detailed insights into air quality trends. Additionally, the application could expand its scope by integrating more robust and diverse APIs to cover a wider range of environmental data, such as weather or pollen levels.

The project could also benefit from improved hosting solutions, offering better performance, faster load times, and enhanced reliability. A shorter and more memorable domain name would also contribute to a better user experience. Other possible improvements include implementing user-specific features, like personalized alerts, and exploring mobile app development to increase accessibility and user engagement. These advancements would make AirSapa more versatile and impactful, aligning with its goal of promoting environmental awareness and public health.