**Air Quality Index (AQI) Forecasting**

**Proof of Concept Design Proposal**

**Prepared for: [Client Name]**  
**Prepared by: [lmw]**  
**Date: April 23, 2025**

# 1. Executive Summary

To present this design proposal for a Proof of Concept (PoC) to deliver a scalable, secure, and efficient Air Quality Index (AQI) forecasting system for US cities. Leveraging publicly available scientific datasets from the National Oceanic and Atmospheric Administration (NOAA) and OpenAQ, our solution uses advanced machine learning (ML) models to provide 24-hour AQI forecasts, adhering to the U.S. Environmental Protection Agency (EPA) standards. The system supports two user types—enterprise and individual—with tailored outputs, including visual content for individual users.

Our proposed architecture is built on Amazon Web Services (AWS), utilizing services such as Amazon S3, SageMaker, RDS, Lambda, EC2, and API Gateway to ensure scalability, security, and cost-efficiency. This document outlines the system architecture, API design, implementation details, benefits, and next steps to move forward with the PoC.

# 2. Project Overview

The objective of this PoC is to develop a cloud-based AQI forecasting system that meets the following requirements:

**Data Sources**: Utilize NOAA Global Surface Summary of the Day and OpenAQ datasets for US cities.

**Machine Learning**: Employ multiple ML models to predict AQI for the next 24 hours, following EPA AQI definitions.

**User Types**: Support enterprise users (requiring raw AQI data) and individual users (requiring AQI data with city-specific images).

**Output**: Provide AQI forecasts with health information and, for individual users, a visual representation reflecting city characteristics and AQI status.

The PoC aims to demonstrate the feasibility of a production-ready system, leveraging AWS’s managed services to replace the current local setup (Django, Next.js, MinIO, PostgreSQL) with a scalable cloud solution.

# 3. Proposed Solution

## 3.1 System Architecture

The proposed architecture could be a serverless or EC2, cloud-native solution hosted on AWS, designed for scalability, security, and maintainability. Key components include:

- Frontend: A Next.js application hosted on Amazon ECS/Fargate, serving AQI forecasts and images to individual users.

- Backend: AWS Lambda functions handle data processing, model training, and forecasting, exposed via Amazon API Gateway.

- Storage: Amazon S3 stores raw datasets, merged data, ML models, and generated images.

- Database: Amazon RDS (PostgreSQL) stores metadata for datasets, models, and images.

- Machine Learning: Amazon SageMaker manages AutoML training and real-time inference for AQI predictions.

- Security: Amazon Cognito provides user authentication and authorization.

- Monitoring: Amazon CloudWatch tracks logs and metrics.

The architecture diagram (Appendix 6.1) illustrates the data flow and component interactions.

## 3.2 AWS Services

The following AWS services are integral to the solution:

1. Amazon S3:

- Stores NOAA and OpenAQ datasets, merged data, trained models, and generated images.

- Features versioning, lifecycle policies, and encryption for data durability and cost optimization.

2. Amazon SageMaker:

- Facilitates AutoML training of ML models (e.g., RandomForest, XGBoost) and hosts endpoints for real-time AQI forecasting.

- Supports custom training scripts and model monitoring.

3. Amazon RDS (PostgreSQL):

- Manages metadata for datasets, models, and images.

- Offers automated backups, Multi-AZ deployment, and encryption.

4. AWS Lambda / EC2:

- Executes backend logic for data ingestion, processing, training orchestration, and forecasting.

- Scales automatically and integrates with other AWS services.

5. Amazon API Gateway:

- Exposes RESTful APIs for enterprise and individual users.

- Supports authentication, rate limiting, and CORS.

6. Amazon ECS/Fargate:

- Hosts the Next.js frontend for individual users.

- Provides serverless container management and auto-scaling.

7. Amazon Cognito:

- Manages user authentication and authorization for secure API access.

- Supports user pools and role-based access control.

8. Amazon CloudWatch:

- Monitors application performance, logs, and metrics.

- Enables proactive issue detection with alarms and dashboards.

9. Amazon SNS/SES:

- Sends notifications (SNS) and emails (SES) for training job status and AQI alerts.

10. Amazon VPC:

- Secures network communication within private subnets.

- Uses VPC endpoints for S3 and SageMaker access.

## 3.3 API Design

The system provides six RESTful APIs, exposed via Amazon API Gateway, to support data ingestion, processing, training, and forecasting:

1. \*/fetch-noaa\*\* (GET): Fetches NOAA data for specified cities and years, stores in S3, and saves metadata to RDS.

2. \*\*/fetch-openaq\*\* (GET): Fetches OpenAQ data, processes sensor data, stores in S3, and saves metadata.

3. \*\*/prepare-training\*\* (POST): Merges NOAA and OpenAQ data, calculates AQI, and stores training/validation/test sets in S3.

4. \*\*/train-model\*\* (POST): Triggers a SageMaker Training Job to train ML models and stores artifacts in S3.

5. \*\*/forecast-aqi\*\* (POST): Generates a 24-hour AQI forecast using a SageMaker Endpoint.

6. \*\*/forecast-aqi-withimg\*\* (POST): Generates an AQI forecast and a city-specific image for individual users.

Detailed API specifications, including parameters, responses, and error handling, are provided in Appendix 6.2.

# 4. Implementation Details

## 4.1 Data Ingestion

- Process: Lambda functions (`/fetch-noaa`, `/fetch-openaq`) retrieve NOAA and OpenAQ data from public S3 buckets (`noaa-gsod-pds`, `openaq-data-archive`). Data is processed (e.g., feature engineering, sensor merging) and stored in S3 buckets (`noaa-data`, `openaq-data`).

- Metadata: Stored in RDS tables (`noaa\_data`, `openaq\_data`) for tracking filenames, buckets, and record counts.

## 4.2 Data Preparation

- Process: The `/prepare-training` Lambda function merges NOAA and OpenAQ datasets, calculates AQI per EPA standards, and splits data into training (Jan-Oct), validation, and test (Nov-Dec) sets. Results are stored in S3 (`merged-data`).

- Metadata: Stored in RDS (`merged\_data` table) with references to training and test filenames.

- Validation: Ensures pollutant data (e.g., PM2.5) is present and correctly formatted.

## 4.3 Model Training

- Process: The `/train-model` Lambda function initiates a SageMaker Training Job, using AutoML to train models like RandomForest and XGBoost on S3 data. Trained models are stored in S3 (`models`).

- Metadata: Stored in RDS (`trained\_models` table) with model paths and leaderboard metrics.

- Optimization: SageMaker’s hyperparameter tuning and model selection ensure high accuracy.

## 4.4 AQI Forecasting

- Process: The `/forecast-aqi` Lambda function invokes a SageMaker Endpoint to predict AQI based on input features (e.g., DEWP, PM2.5). Results include predicted AQI, EPA category, and color code.

- Delivery: Returned via API Gateway for enterprise users or rendered in the Next.js frontend for individual users.

## 4.5 Image Generation

- Process: The `/forecast-aqi-withimg` Lambda function generates AQI forecasts and city-specific images (e.g., via a third-party API or SageMaker model). Images are stored in S3 (`generated-images`).

- Delivery: Image URLs are returned to the Next.js frontend for individual users, reflecting city characteristics and AQI status.

## 4.6 Frontend Delivery

- Process: The Next.js application, hosted on ECS/Fargate, fetches AQI forecasts and images via API Gateway. Static assets are served from S3 via CloudFront.

- Security: Cognito authenticates users, ensuring secure access to APIs and images.

# 5. Benefits of the Proposed Solution

- Scalability: Serverless components (Lambda, API Gateway, Fargate) and managed services (S3, RDS, SageMaker) scale automatically with demand.

- Cost-Efficiency: Pay-per-use pricing for Lambda, S3, and SageMaker, with lifecycle policies for data archiving.

- Maintainability: Managed services reduce operational overhead, allowing focus on feature development.

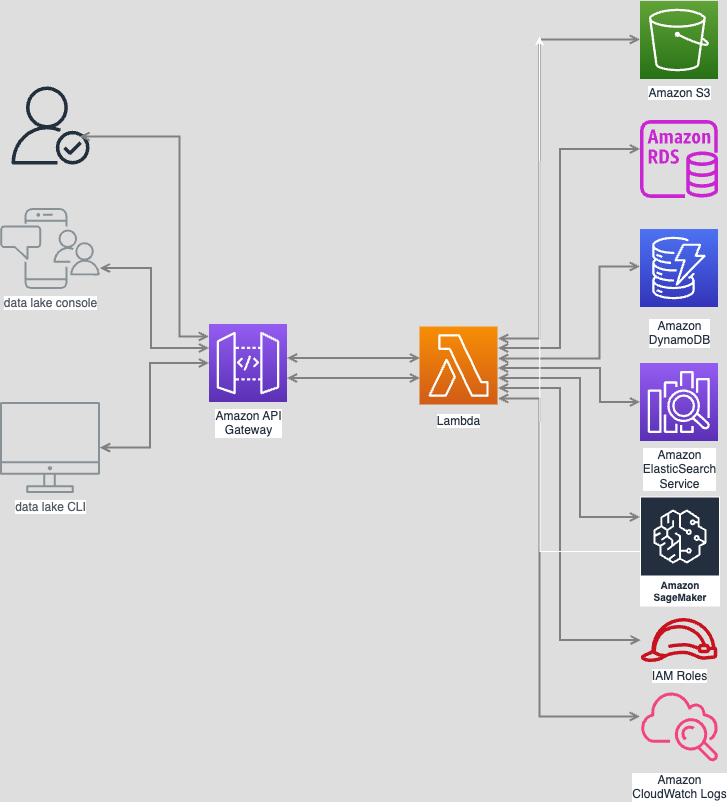
- Flexibility: Supports both enterprise and individual users with tailored outputs.

-Accuracy: SageMaker’s AutoML ensures robust ML models, validated against EPA AQI standards.

- User Experience: Next.js frontend delivers responsive, visually appealing AQI forecasts for individual users.

# 6. Appendix

## 6.1 Architecture Diagram



## 6.2 API Specifications

### API 1: /fetch-noaa

- \*\*Method\*\*: GET

- \*\*Parameters\*\*:

- `city\_name` (string, required): US city name.

- `year\_start` (integer, optional, default=2023): Start year.

- `year\_end` (integer, optional, default=2023): End year.

- \*\*Response\*\* (200):

```json

{

"results": [

{

"status": "success",

"message": "Successfully processed NOAA data for New York",

"city": "New York",

"s3\_file": "noaa\_725030\_New\_York\_2023-2023.csv",

"noaa\_data": {

"filename": "noaa\_725030\_New\_York\_2023-2023.csv",

"bucket": "noaa-data",

"city\_name": "New York",

"station\_id": "725030",

"year\_start": 2023,

"year\_end": 2023,

"record\_count": 365

}

}

],

"total\_processed": 1,

"successful": 1,

"failed": 0

}

```

- \*\*Error\*\* (400/500): `{"error": "Invalid city\_name"}`

### API 2: /fetch-openaq

- \*\*Method\*\*: GET

- \*\*Parameters\*\*:

- `city\_name` (string, required): US city name.

- `parameter` (string, optional, default='pm25'): Pollutant type.

- `year\_start` (integer, optional, default=2023): Start year.

- `year\_end` (integer, optional, default=2023): End year.

- `radius\_meters` (integer, optional, default=16100): Data radius.

- \*\*Response\*\* (200):

```json

{

"results": [

{

"status": "success",

"message": "Successfully processed data for New York",

"city": "New York",

"s3\_file": "openaq\_New\_York\_2023\_12345.csv"

}

],

"total\_processed": 1,

"successful": 1,

"failed": 0

}

```

- \*\*Error\*\* (400/500): `{"error": "No data found"}`

### API 3: /prepare-training

- \*\*Method\*\*: POST

- \*\*Parameters\*\*:

- `noaa\_filename` (string, required): NOAA data S3 key.

- `openaq\_filename` (string, required): OpenAQ data S3 key.

- `pollutant` (string, required): Pollutant (e.g., pm25).

- \*\*Response\*\* (200):

```json

{

"train\_filename": "train\_pm25\_noaa\_725030\_New\_York.csv",

"test\_filename": "test\_pm25\_noaa\_725030\_New\_York.csv",

"bucket": "merged-data",

"train\_record\_count": 300,

"test\_record\_count": 65

}

```

- \*\*Error\*\* (400/500): `{"error": "Pollutant not found"}`

### API 4: /train-model

- \*\*Method\*\*: POST

- \*\*Parameters\*\*:

- `train\_filename` (string, required): Training data S3 key.

- `merged\_data\_id` (integer, required): Merged dataset ID.

- `model\_name` (string, optional, default='aqi\_predictor'): Model name.

- `time\_limit` (integer, optional, default=3600): Training time (seconds).

- `models\_to\_train` (list, optional, default=['RF', 'XT', 'XGB', 'GBM']): Models to train.

- \*\*Response\*\* (200):

```json

{

"model\_id": 1,

"model\_name": "aqi\_predictor",

"model\_path": "models/aqi\_predictor\_20250423\_100000.tar.gz",

"leaderboard": {

"model": ["RandomForestMSE", "XGBoost"],

"score\_val": [0.85, 0.82]

},

"zip\_size\_bytes": 5242880

}

```

- \*\*Error\*\* (400/500): `{"error": "Missing columns"}`

### API 5: /forecast-aqi

- \*\*Method\*\*: POST

- \*\*Parameters\*\*:

- `model\_id` (integer, required): Trained model ID.

- `city\_name` (string, required): US city name.

- `forecast\_data` (object, required): Features (DEWP, WDSP, MAX, MIN, PRCP, MONTH, pm25\_value).

- \*\*Response\*\* (200):

```json

{

"city\_name": "New York",

"forecast": {

"date": "2025/04/24",

"city\_name": "New York",

"pm25\_value": 12.5,

"predicted\_aqi": 50.0,

"calculated\_aqi": 52.0,

"descriptor": "Good",

"color": "#00E400"

}

}

```

- \*\*Error\*\* (400/500): `{"error": "Missing features"}`

### API 6: /forecast-aqi-withimg

- \*\*Method\*\*: POST

- \*\*Parameters\*\*: Same as /forecast-aqi.

- \*\*Response\*\* (200):

```json

{

"city\_name": "New York",

"forecast": {

"date": "2025/04/24",

"city\_name": "New York",

"pm25\_value": 12.5,

"predicted\_aqi": 50.0,

"calculated\_aqi": 52.0,

"descriptor": "Good",

"color": "#00E400"

},

"image\_url": "https://generated-images.s3.us-east-1.amazonaws.com/new\_york\_20250424.jpg"

}

```

- \*\*Error\*\* (400/500): `{"error": "Image generation failed"}`