**SE 4485.001 -**

**Software Engineering Project**

**Software Requirements Specification (SRS)**

**City Level Air Quality Prediction**

**Group 1 -**

**Raytheon (Team A):**

**Jay Chung <cwc130330> Amelia Quinn <qcb220000>**

**Kevin Melo <ksm220005> AJ Kimbrough <ank210005>**

**David Santos <des210001> Andrew Enright <ame210008)>**

**In collaboration with:**

**Marc Perna <**[**marc.perna@rtx.com**](mailto:marc.perna@rtx.com)**>,**

**Trevor Lang <**[**trevor.a.lang@rtx.com**](mailto:trevor.a.lang@rtx.com)**>,**

**Ryan Havens <**[**ryan.havens@rtx.com**](mailto:ryan.havens@rtx.com)**>,**

**Trey Williams <**[**trey.williams@rtx.com**](mailto:trey.williams@rtx.com)**>**

**09/SEP/2025**

**Version 0.0.3**

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **DATE:** | **VERSION:** | **DESCRIPTION:** | **AUTHOR(S):** |
| **30/AUG/2025** | **0.0.1** | **Created: Interim Draft,  Added: Team Members** | **cwc130330** |
| **09/SEP/2025** | **0.0.2** | **Created: Functional and Nonfunctional Requirements Revised: Grammar, Introduction, Purpose** | **qcb220000** |
| **10/SEP/2025** | **0.0.3** | **Added: Abbreviations & Definition** | **cwc130330** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**TABLE OF CONTENTS**

1. **INTRODUCTION 4**

**1.1 PURPOSE 5**

**1.2 SCOPE 5**

**1.3 DEFINITIONS, ACRONYMS & ABBREVIATIONS 5**

**1.4 REFERENCES 5**

1. **OVERALL DESCRIPTION 6**

**2.1 PRODUCT PERSPECTIVE 6**

**2.2 PRODUCT FUNCTIONS 6**

**2.3 USER CHARACTERISTICS 6**

**2.4 CONSTRAINTS 6**

**2.5 ASSUMPTIONS & DEPENDENCIES 6**

1. **SPECIFIC REQUIREMENTS 7**

**3.1 FUNCTIONAL REQUIREMENTS 7**

**3.2 NON-FUNCTIONAL REQUIREMENTS 7**

**3.3 EXTERNAL INTERFACE REQUIREMENTS 7**

1. **SYSTEM ARCHITECTURE 8**
2. **SUCCESS CRITERIA 9**
3. **RISKS 9**

**Software Requirements Specification (SRS)**

1. **INTRODUCTION**

We are a team of software engineers developing a predictive analytics application that predicts air quality index for a city based on historic AQI data and weather forecasts.

Team Name: Group 1 - Raytheon (Team A)

Project Name: City Level Air Quality Prediction

Team Website Link: TBD

Team Members:

1. Jay Chung (cwc130330) - Team leader, Software & AI Engineer
2. Amelia Quinn (qcb220000) - Software & AI Engineer
3. AJ Kimbrough (ank210005) - Lead Architect, Software & AI Engineer
4. Kevin Melo (ksm220005) - Software & AI Engineer
5. David Santos (des210001) - Software & AI Engineer
6. Andrew Einright (ame210008) - Software & AI Engineer

|  |  |  |  |
| --- | --- | --- | --- |
| DELIVERABLE: | DUE DATE: | RESPONSIBILITY: | TOOLS: |
| Weekly Report | Every Friday | Team leader | Google forms, Outlook Email. |
| Project Management Plan | 09/12/2025 | Team leader, Lead Architect | TBD |
| Requirements Documentation | 09/26/2025 | Team leader, TBD | TBD |
| Architecture Documentation | 10/24/2025 | Team leader, TBD | TBD |
| Detailed Design Documentation | 11/07/2025 | Team leader, TBD | TBD |
| Test Plan | 11/21/2025 | Team leader, TBD | TBD |
| Final Project Presentation Slides | 12/02/2025 | Team leader, TBD | TBD |
| Final Project Report | 12/05/2025 | Team leader, TBD | TBD |

**1.1 PURPOSE**

This project aims to develop an application designed to use prior data regarding air pollutants correlated with past weather data to predict air quality indexes based on projected future weather data.

**1.2 SCOPE**

The system will:

* Collect historical AQI data from the U.S. EPA AQS.
* Retrieve daily weather forecasts from the NWS.
* Process and join data into a predictive feature set.
* TBD

**1.3 DEFINITIONS, ACRONYMS & ABBREVIATIONS**

Definitions:

* Predictive Analytics Model: Models that are designed to assess historical data, discover patterns, observe trends, and use that information to predict future trends.

Acronyms:

* TBD

Abbreviations:

* AQI: Air Quality Index
* AQS: Air Quality System
* EPA: U.S. Environmental Protection Agency
* NWS: National Weather Service
* TBD

**1.4 REFERENCES**

* EPA AQS data: [aqs.epa.gov](http://aqs.epa.gov), <https://www.epa.gov/outdoor-air-quality-data>
* NWS API: [api.weather.gov](http://api.weather.gov)
* AirNow AQI Scale: [airnow.gov/aqi](http://airnow.gov/aqi)

1. **OVERALL DESCRIPTION**

**2.1 PRODUCT PERSPECTIVE**

* The system runs locally on a personal computer, optionally in a Docker container.
* It integrates data from EPS and NWS APIs, storing results in SQL.
* It provides a dashboard (e.g. Streamlit or React).
* TBD

**2.2 PRODUCT FUNCTIONS**

* Data ingestion from EPA AQS (e.g. historical data) and NWS (e.g. forecast data).
* Data processing and feature construction
* TBD

**2.3 USER CHARACTERISTICS**

* Primary Users: Students, project sponsors (e.g. Raytheon).
* Secondary Users: Users with basic computer literacy.

**2.4 CONSTRAINTS**

* Must run in <5 minutes on commodity hardware.
* Must be reproducible via environment files.

**2.5 ASSUMPTIONS & DEPENDENCIES**

* APIs are stable, accessible, and available for use.
* Forecast and AQI data will be available daily.
* Internet availability is required for data retrieval.
* TBD

1. **SPECIFIC REQUIREMENTS**
   1. **FUNCTIONAL REQUIREMENTS**

* Data ingest (history): System shall fetch daily AQI summaries for the chosen locality from [EPA AQS/AirData](https://aqs.epa.gov/aqsweb/airdata/download_files.html) (CSV).
* Data ingest (forecast): System shall fetch [NWS forecast](https://www.weather.gov/documentation/services-web-api) JSON for the locality on a daily basis.
* Processing: System shall clean/join AQI & forecast data and construct features (recent AQI lags + basic forecast attributes).
* Storage: System shall persist raw and processed datasets in a relational store (SQLite) with simple indices on date.
* Modeling: System shall train at least one classification model and expose a predict-tomorrow function.
* Visualization: System shall render (a) 30-day AQI history, (b) tomorrow’s predicted category, and (c) model explainers.
* Automation: Single command to refresh data and re-train on a schedule (e.g., weekly CRON/GitHub Action).
* Documentation & handoff: README + user guide + final presentation as requested
* TBD

**3.2 NON-FUNCTIONAL REQUIREMENTS**

* Simplicity: End-to-end refresh completes in < 5 minutes on a student laptop (city-level scope).
* Reproducibility: Dev environment pinned via requirements.txt (or conda) and a Makefile.
* Reliability: If an API call fails, system retries and logs an actionable message.
* Transparency: Model choice, features, and limitations documented plainly (no black box claims).
* Data ethics/usage: Respect EPA/NWS terms; cite sources in UI/README. National Weather Service: docs.airnowapi.org.
* Portability: Runs locally via python -m app and optionally containerized (Dockerfile).
* TBD

**3.3 EXTERNAL INTERFACE REQUIREMENTS**

* TBD

1. **SYSTEM ARCHITECTURE**

* TBD

1. **SUCCESS CRITERIA**

* Model performance exceeds the majority-class baseline by ≥10 points in macro-F1.
* Dashboard displays correct pipeline status and results.
* Refresh completes in <5 minutes on a median-power consumer laptop.
* Codebase includes setup instructions, unit tests, and user documentation.
* TBD

1. **RISKS**

* Data Gaps: Not all pollutants are reported daily; migrate using AQI category fallback.
* Forecast Variability: Textual forecasts vary; prioritize numeric features when possible.
* Over-Scoping Risk: Limit to a single city for MVP.
* TBD