Projet UML : AirWatcher

INITIALIZATION DOCUMENT

Introduction:

The growing concern for environmental sustainability has led to an increased emphasis on monitoring and improving air quality. Recognizing this imperative, we are making a project to develop a state-of-the-art software application, AirWatcher. This application is commissioned by a government environmental protection agency to facilitate the monitoring and analysis of air quality across a large territory.

AirWatcher is designed to process and analyze environmental data collected through an array of sensors dispersed throughout the region. By accurately assessing air quality metrics, AirWatcher will enable the agency to maintain sensor integrity, manage air cleaners, and contribute effectively to environmental health policies. It will empower the agency with data-driven insights to implement measures for improving air quality and offer a tangible impact on public health and ecological balance.

Planning:

Team: binome B3120+ binome B3123

Roles:

- Adrien HOUDOUX : C++ programmer
- Mohamed FAKRONI : document manager / programmer
- Youssef CHAOUKI : Data scientist/
- Lizhi ZHANG : System Architect

Organization:

03/04/2024 Initialization document

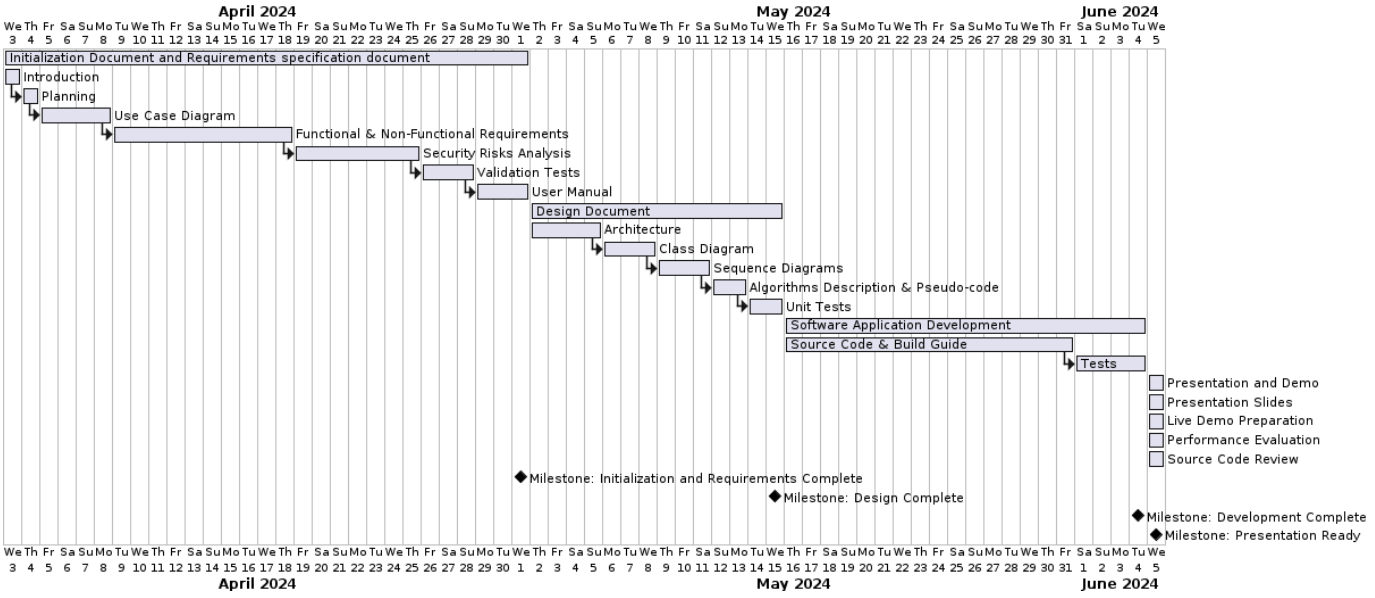
02/05/2024 Design document

07/05/2024,29/05/2024,05/06/2024 Software application development

- Source code + guide to build the executable

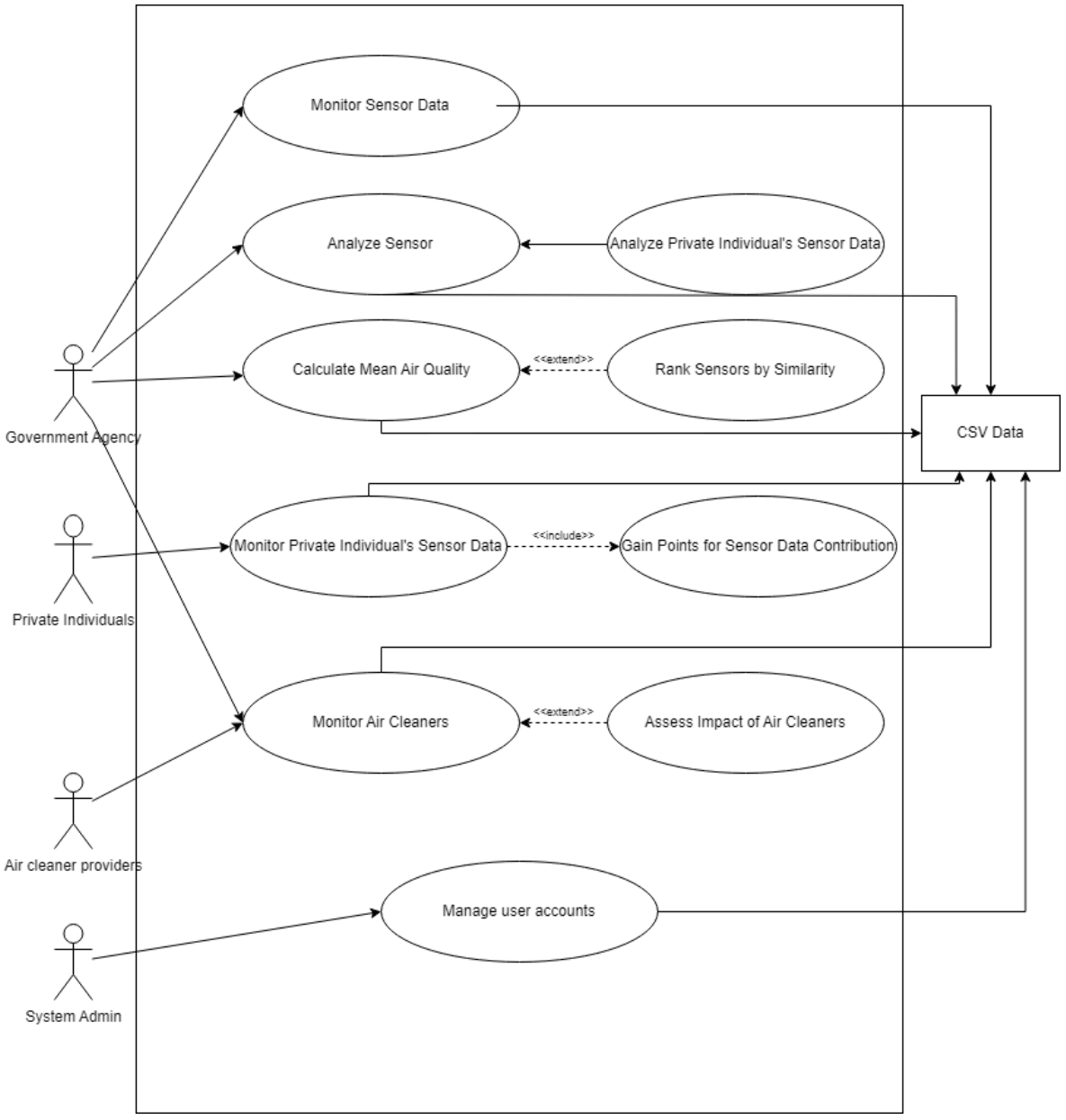
- Tests (source code + data + any non-regression procedures)

Gantt chart:


April 2024
May 2024
June 2024
Initialization Document and Requirements specification document
Introduction
Planning
Use Case Diagram
Functional & Non-Functional Requirements
Security Risks Analysis
Validation Tests
User Manual
Design Document
Architecture
Class Diagram
Sequence Diagrams
Algorithms Description & Pseudo-code
Unit Tests
Software Application Development
Source Code & Build Guide
Tests
Presentation and Demo
Presentation Slides
Live Demo Preparation
Performance Evaluation
Source Code Review
Milestone: Initialization and Requirements Complete
Milestone: Design Complete
Milestone: Development Complete
Milestone: Presentation Ready

REQUIREMENTS SPECIFICATION DOCUMENT

I. Use case diagram



II. Functional and non-functional requirements of the system

I. Functional requirements of the system

Function	**User authentication**
Description	Allows the user to sign in to the application using their provided credentials. Displays the menu associated with their privileges
Input	User authentication key
Output	Boolean
Sources	Accounts CSV to validate user input
Action	Validate user input and find associated privilege level
Precondition	Database does not change during execution of the command

Function	**Analysis and comparison of sensor readings**
Description	The application can perform analysis and comparison of sensor readings. Specifically, the application can calculate the mean air quality of a given area (even if no sensors are present), estimate the improvement over time and identify areas with similar readings. The application will provide an interface for user to input parameters and visualize the results
Input	Area analysis : - Geographical area - Time frame Similarity analysis : - Reference sensor - Time frame
Output	Data visualizations : plots, readings
Sources	Sensor readings CSV
Action	Perform requested calculations and analysis
Precondition	Readings are present in the database.

Function	**User and sensor management**
Description	Allows administrators to manage users, by adding, deleting, and modifying user information. Allows users to manage sensors associated with their account, including adding and removing sensors. Rewards users whose sensors provide data
Input	User information, data
Output	None
Sources	Accounts CSV
Action	Modify user information, modify sensors associated to user, reward users
Precondition	Database does not change during execution of the command
Postcondition	User information is modified in the database

Function	**Detection of faulty or malicious behavior**
Description	The application monitors sensor data to detect faulty or malicious behavior (such as sending false data) and labels data from suspicious sensors as unreliable, excluding them from further analysis.
Input	None
Output	List of potentially malicious sensors to be flagged as faulty
Sources	Sensor readings CSV, Sensors CSV
Action	Exclude data from future analysis if faulty or malicious behavior is detected
Precondition	Database does not change during execution of the command,
Postcondition	Affected sensors are flagged as excluded in the database

II. Non-functional requirements of the system

Product Requirements:
Usability Requirements : The system must have different interfaces for companies and individuals, ensuring ease of use and accessibility.

Efficiency Requirements:	
Performance Requirements:	Space Requirements:
Algorithms used for analyzing sensor data, calculating air quality indices, and comparing sensors must be optimized for speed. No request should take more than 3000 ms to complete.	The system must have enough storage space to save historical air quality data collected from various sensors. This includes immediate data storage needs and projected growth over time. We advocate for at least 1TB of mass storage.

Dependability Requirements:
The system should be accessible when needed, implying a high uptime and minimal downtime for maintenance or due to system failures. A minimum of 99% uptime is required.

Security Requirements:
Protection of sensitive data, including the locations of air cleaners and personal sensors, against unauthorized access or tampering. Secure identification of users to prevent unauthorized access to the application's functionalities.

Organizational Requirements:		
Organizational Requirements	Operational Requirements	Development Requirements
The application should be developed and operated with a consideration for environmental sustainability.	The system must be reliable and robust, with minimal downtime, as it will be used for monitoring critical environmental parameters. Clear procedures should be in place for regular maintenance and updates to the application, ensuring it continues to function correctly and securely. There must be robust data backup and recovery solutions to prevent data loss and to ensure continuity in the event of a system failure.	The application should undergo thorough testing, including unit, integration, and performance tests, to ensure it meets functional and non-functional requirements. Comprehensive documentation must be produced, including technical specifications, user manuals, and developer guides.

External Requirements			
Regulatory Requirements	Ethical Requirements	Legislative Requirements	
Since the system handles data that may be considered personal (e.g., locations of private sensors), it must comply with relevant data protection laws.	The algorithms used for data analysis and the decision-making processes should be transparent, especially when they affect environmental policies.	Accounting Requirements : The system should be able to track and report on the use of resources, possibly including the operational costs of sensors and air cleaners, to assist with financial planning and budgeting.	Safety & Security Requirements The system must ensure that the operation of air cleaners is monitored and reported safely without causing harm to the public or the environment.

III. Analysis of security risks

Valuable data

- Sensor data
- Sensors and cleaners positions
- Application functionality
- Server resources

Security risks

- Unauthorized access to data
- Data integrity compromise
- Denial of service

Potential attackers

- Malicious private individuals
- Competitors
- Malicious groups
- Malicious insiders

Vulnerabilities

- Insufficient strength/frequency of the false data verification
- Lack of input validation
- Lack of anti DoS measures
- Weak user authentication
- Misuse of privileges from insiders

Consequences

- Unauthorized access to valuable data
- Server overload with excessive requests
- False data uploaded to the server from corrupt sensors
- Application unavailability
- Loss of trust from the customers
- Inaccurate analysis and potential tampering with decision-making

Counter-measures

- Strong authentication mechanisms
- Strong and frequent data integrity checks
- DoS protection with traffic limiting
- Secure coding practices and input validation
- Employee monitoring

IV. Validation tests

Analysis and Comparison of Sensors :

- Verify Detection of Faulty Sensors:

  - Test: Introduce sensor data with no faults.
    Expected Outcome: System does not report that the sensor is faulty, nor does it flag the readings.

  - Test: Introduce sensor data with known faults.
    Expected Outcome: System identifies and reports the faulty sensors.

- Verify Anomaly Detection in Sensor Measurements:

  - Test: Inject abnormal sensor readings into the system.
    Expected Outcome: System detects anomalies and flags the abnormal readings.

  - Test: Inject normal sensor readings into the system.
    Expected Outcome: System does not trigger any alert.

- Verify Sensor Data Comparison with Reference Sensor:

  - Test: Select a reference sensor and compare its data with others.
    Expected Outcome: System accurately calculates the similarity between the reference sensor and others.

Calculation or Estimation of Air Quality :

- Verify Calculation of Air Quality at the specific location:

  - Test: Specify a geographic area without sensor coverage.
    Expected Outcome: System predicts air quality using interpolation

  - Test: Specify a geographic area with a sensor at the exact sensor position.
    Expected Outcome: System returns the corresponding data for the given time directly from the data stored by the sensor.

- Verify Visualization of Air Quality Results:

  - Test: Input geographic and time parameters for air quality calculation.
    Expected Outcome: System generates visual representations of air quality trends that match the input parameters.

- Verify Consistency of Air Quality Estimations:

  - Test: Input the same geographic area and time period multiple times.
    Expected Outcome: System consistently provides the same air quality estimation for identical inputs.

Data Aggregation :

- Verify Calculation of Air Quality Statistics:

  - Test: Provide sensor data for a specific time period.
    Expected Outcome: System accurately calculates metrics

- Verify Calculation of the mean Air Quality

  - Test: Specify a circular area.
    Expected Outcome: System accurately calculates mean and displays the result.

  - Test: Specify a circular area and a specific time period.
    Expected Outcome: System accurately calculates mean and displays the result for the corresponding period.

User and Sensor Management :

- Verify User Information Modification by Administrators:

  - Test: Attempt to modify user information with administrative privileges.
    Expected Outcome: System allows administrators to successfully update user information.

- Verify Addition of Sensors by Users:

  - Test: Allow users to add sensors to their accounts.
    Expected Outcome: System permits users to add sensors, updating the database accordingly.

- Verify Removal of Sensors by Users:

- Test: A user removes a sensor from their account.
  Expected Outcome: The system successfully deletes the sensor associated with the user's account without affecting other functionalities.

- Verify User Access Control:

- Test: Attempt to access administrative functions with standard user credentials.
  Expected Outcome: System restricts access to administrative features for non-administrative users.

Detection of Malicious Behavior :

- Verify Monitoring and Detection of Malicious Sensor Behavior :

- Test: Inject false sensor data to mimic malicious behavior.
  Expected Outcome: System identifies and marks the data from suspicious sensors as unreliable.

- Test: Inject normal sensor data to mimic normal user behavior.
  Expected Outcome: System does not trigger any alert.

- Verify Exclusion of Suspect Sensor Data from Analysis :

- Test: Mark data from suspect sensors as unreliable.
  Expected Outcome: System excludes unreliable data from further analysis and ensures it does not impact subsequent results.

- Test: Execute analysis of the data of an area where a trusted individual is.
  Expected Outcome: System uses data from the individual to execute the analysis.

V. User manual

Introduction

Welcome to AirWatcher, a software application designed to empower government agencies in monitoring and analyzing air quality data. This user manual guides you through AirWatcher's functionalities, tailored to different user roles.

User Roles and Access

AirWatcher caters to different user roles with varying levels of access:

- **Government Agent:** Full access to monitor data, analyze sensor performance, and evaluate air quality across locations.
- **Air Cleaner Provider:** Access to observe the effectiveness of air cleaners in improving air quality.
- **Private Individual:** Monitor data from your personal sensors and track points earned for contributing data.

Getting Started

System Requirements: You likely meet them if you're running Linux already (most distributions come with a modern processor and enough RAM).
Installation: The GNU C++ compiler (g++) is often pre-installed. Verify with g++ --version in your terminal. If not installed, use your package manager (e.g., sudo apt install build-essential on Ubuntu/Debian).

To launch the application and sign in:

1. Execute the AirWatcher application from the designated directory on the server.
2. Enter your credentials provided by the system administrator.
3. Upon successful authentication, you will gain access to the AirWatcher console interface.

Main Features

Government Agent

- **Analyze Sensor Data:** View real-time and historical data from all sensors, including sensor ID, location, timestamps, and measured attributes (e.g., O3 concentration).
- **Calculate Mean Air Quality:** Assess average air quality for a specified geographical area and time period. Users can define the area by radius around a central point or specify a specific latitude and longitude. The application calculates the mean Air Quality Index (AQI) based on sensor data within the chosen area and timeframe.
- **Compare Sensor Similarity:** Rank all sensors based on their similarity to a chosen sensor during a specified period. This helps identify sensors with similar air quality patterns, potentially indicating similar environmental conditions.

- **Evaluate Sensor Performance:** Analyze sensor data for consistency and identify potential malfunctions. The application uses statistical methods to flag sensors with readings deviating significantly from the expected range.

Air Cleaner Provider

- **Observe Air Cleaner Impact:** Analyze the impact of air cleaners on air quality within a specified area. Users can define the area and timeframe to assess changes in AQI before, during, and after air cleaner operation. This helps evaluate the effectiveness of air cleaners in improving air quality.

Private Individual

- **Monitor Sensor Data:** View real-time and historical data from your personal sensor, including timestamps, measured attributes, and the corresponding AQI value.
- **Track Data Points:** Keep track of points earned for contributing data to AirWatcher. Points are awarded each time your sensor data is used in a query by any authorized user.

Utilizing the Console Interface

The console interface presents a menu with options tailored to your user role. Use the corresponding numbers or keywords to select the desired functionality. Each option might prompt for additional information, such as sensor IDs, geographical coordinates, or timeframes. The application will display the results of your queries on the console screen.

Data Access

While users cannot directly access the raw data files, AirWatcher provides functionalities to view and analyze data relevant to your role. Government agents have the most extensive access for comprehensive analysis, while Air Cleaner Providers and Private Individuals can access data related to their specific areas of interest.

Conclusion

AirWatcher empowers government agencies and stakeholders with valuable tools for air quality monitoring and analysis. This user manual provides a basic overview. More detailed information on specific functionalities and data interpretation is available within the application through help menus or command descriptions.