Information System of Air Quality Assessment Based of Ground Stations and Meteorological Data Monitoring

Bohdan Molodets[a], Volodymyr Hnatushenko[b], Daniil Boldyriev[a], Tetiana Bulana[b]

[a] *Oles Honchar Dnipro National University, 35 av. Dmytra Yavornytskoho Dnipro, 49044, Ukraine*
[b] *Dnipro University of Technology, 19 av. Dmytra Yavornytskoho, Dnipro, 49005, Ukraine*

Abstract

Monitoring ground stations and collecting meteorological data are essential solutions for assessing air quality. A developed information system can aggregate and process the data obtained. The data is transformed into a unified format and returned through written application programming interfaces (APIs). Client interfaces were created for convenient display of the results. The project infrastructure is designed for easy deployment. The architectural solution for creating the system proposes a toolkit that optimizes system operation when performing complex tasks through asynchronous execution. The use of Docker during deployment provides additional capabilities. To calculate the distribution of emissions in Kryvyi Rih, the CALPUFF model was employed for data processing. The article describes the client part structure and interface description. It also displays the processed data, which is the result of applying a mathematical model to the meteorological and station data.

Keywords

Information system, air quality monitoring, docker, CALPUFF, image processing, data visualization

1. Introduction

From smog hanging over cities to smoke inside the home, air pollution poses a major threat to health and climate. Ambient (outdoor) air pollution in both cities and rural areas is causing fine particulate matter which result in strokes, heart diseases, lung cancer, acute and chronic respiratory diseases. Additionally, around 2.4 billion people are exposed to dangerous levels of household air pollution, while using polluting open fires or simple stoves for cooking fueled by kerosene, biomass (wood, animal dung and crop waste) and coal. Moreover mining-related deforestation destroys carbon storage capacities, with implications for air quality [1]. The combined effects of ambient air pollution and household air pollution is associated with 7 million premature deaths annually [2].

It is necessary to develop the information system, which solve this problem by sending warnings about danger to people. System of air quality monitoring is the system, which visualize current air quality state in real time, save historical information to storage and build pollution forecasts. It is important to collect data from opensource resources or receive data straightly from ground station for detecting potential hazardous zones. Remote sensing also can be used as alternate resource. In our case remote sensing is the processing of satellite imagery, which represents the Earth in different spectral ranges [3, 4].

IntelITSIS'2023: 4th International Workshop on Intelligent Information Technologies and Systems of Information Security, March 22–24 2023, Khmelnytskyi, Ukraine
EMAIL: bogdan.molodets@gmail.com (B. Molodets); vvgnat@ukr.net (V. Hnatushenko); boldyrov@gmail.com (D. Boldyriev); tatyana.bulanaya@gmail.com (T. Bulana).
ORCID: 0000-0002-7802-389X (B. Molodets); 0000-0003-3140-3788 (V. Hnatushenko); 0000-0002-8502-1446 (D. Boldyriev); 0000-0001-6346-3326 (T. Bulana).

The information system has been developed that can fetch, store, process, and display any type of data from any source without losing velocity. The novelty of this article is the methodology developed for creating an information system that utilizes data from various sources such as ground stations or meteorological systems for air quality assessment. The paper combines a mathematical model with system architecture, which is crucial in creating a monitoring information system. The created system is currently used for storing, modeling, and predicting air quality.

2. Related Works

There are many global air quality monitoring services that are designed and operated throughout the world, but most of them are not designed to collect the data needed to estimate population performance on sensitive air pollutants. The problem of air quality monitoring is the subject of many scientific studies. In paper [5], authors provide a comprehensive survey of the state-of-the-art in urban air quality and highlight open issues and research challenges in dealing with urban air pollution. Several studies applied machine learning for air quality mapping and forecasting. In [6] authors applied the machine learning technique with time series of particulate matter pollution records to predict and develop a particulate matter pollution susceptibility map. The random forest was applied to verify and visualize the relationships between the particulate matter and different independent variables. Paper [7] seeks to create a multi-modal machine learning model for predicting air-quality metrics where monitoring stations do not exist. The paper [8] proposes a refined urban air quality estimation method that fuses multisource spatial-temporal data. This method integrates meteorological data with urban social activity data to form a comprehensive environmental data set. The study [9] explores the changes in air quality index during various COVID-19 lockdowns utilizing satellite data. The comparison between modelling tools [10, 11] describes abilities and benefits of use CALLPUFF and other models such as AERMOD and CDF. In paper [12] authors provide results of air quality modeling by using AERMOD Software program (BREEZE AERMOD software): simulation results performed that observed area was facing high concentration CO. Simulation in paper [13] shows dispersion of four pollutants (SO_2, CO, NO_2, and PM2.5) emitted from the Daura refinery in Baghdad during the summer of 2013 and winter of 2013-2014 using the CALPUFF modeling system: hourly emission rates of pollutants based on the actual amounts of fuel consumed at the refinery was calculated.

However, most of the proposed air quality monitoring methodologies need to be further assessed and reviewed before it can be used operationally in other cities.

3. Architecture of system

As the programming language of information system Python was used, as a framework - Django Rest Framework, as DB (database) - PostgresQL. Resources, which contain required data are:

- measuring stations;
- Third-party public aggregators (open databases, web portals);
- satellite data from international space programs (Sentinel-2, Sentinel-3, Landsat-8, etc.).

It is important to create API, that manually run collecting data task, for necessary data aggregating. This solution has several disadvantages: there is not control over the requesting frequency, because that operation is started manually; the more data sources the system has, the harder it is to serialize them (lead to a single format) for storage in the database; the more resources need to "interview", the greater probability of getting system overload; some tasks (for example, downloading/processing satellite images) take a long time period to be finished, and Python programming language is synchronous - waiting for the previous tasks to be completed. This causes inconvenience to the performance of the information system.

Celery's task manager solves this problem by providing the ability to create synchronous/asynchronous tasks in the information system that can be performed in the background, in a different process [14]. Celery provides user interfaces for easy management such as Flower or built-in django-celery. This system consists of task (function to be performed), worker (entity that performs tasks), broker (a system that stores messages in the form of a queue), task producer (task provider).

Celery allows you to run worker-s, brokers in another stream or on another physical machine, because it is a distributed system. It also means that, depending on the needs of the system and in the presence of physical servers, the developer can run N worker, where N is an integral discrete number, which is shown in Figure 1. RabbitMQ can be used as a message broker, which allows you to exchange messages between multiple servers by sending/receiving messages.

Knowing that RabbitMQ is written in the Erlang programming language, Erlang must be installed to the server. To simplify the deployment of the described architecture, it is better to use the virtual container approach - abstraction at the software level, which allows to quickly launch several containers at once in one system. A container is a separate system unit that contains program code with auxiliary packages that allows to deploy the project in different environments. In turn, Docker containers store code, system software, environment settings, etc. [15]. This approach will guarantee the same behavior of the developed system regardless of the system characteristics of the machine (system isolation). Among the advantages, note the possibility of using ready-made images (assembled containers), which facilitates further development, requiring the developer only to configure services (images). As a result, to deploy such a system, it is necessary to add a configuration file (Dockerfile, when using docker-compose - docker-compose.yml) and the presence of the installed Docker client on the working machine.

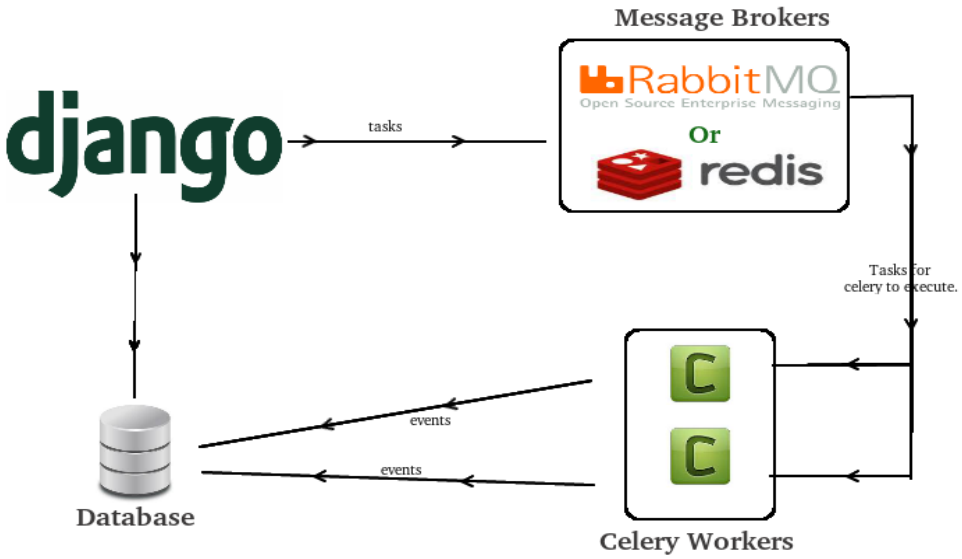

Figure 1. Scheme of the system with the Celery task manager (example with two worker-s) [16]

The software tends to grow, and new team members who have different levels of knowledge should be able to join the team and start working on one part of the program, even if they are not familiar with the whole architecture.

These requirements stipulate that the program must have the following basic functions:

- modular design;
- unidirectional data flow;
- predictable state management;
- communication layer for asynchronous requests;
- presentation layer does mot depends on the base layer.

The development of modular design and project decomposition into smaller parts (modules) is due to the need to be able to easily connect and exclude our modules at any need, without spending a lot of time and without disturbing the project structure, cover each part of the program with separate unit tests and make it possible to avoid conflicts during development, while increasing the number of the team. So, this approach divides the application into blocks of functionality and facilitates its maintenance and testing.

As a framework for the development of the client part, it was decided to choose Angular. Analyzing the results of the research, we can conclude that the speed of these two frameworks is very similar. This

is probably because Vue.js began its existence as a branch of Angular, and it is written in a very similar style to Angular [17]. The main advantage of Vue.js is its performance and low entry threshold. The main factors that influence the choice of Angular as a framework is its structure, the presence of typing support for the above-described components from under the box. Therefore, you should not use Angular in the development of "landing pages", services with a "thin" client, small projects.

The application is divided into three layers - presentational, abstract and main. The presentation layer is used to display HTML and process user interactions. The abstraction layer handles the relationship between the presentation and main layers. The main layer contains the logic of the application core, for example, data manipulation, communication with the API, etc. Each layer is divided into modules, ordered by function. Modules are separate blocks of code that can be connected and excluded as needed. Each block was separated by a module. It was decided to separate asynchronous services (services) and data management into their own modules, since they perform different tasks. The same goes for presentation modules. Presentation modules are separate blocks that are independent of other modules. They include the necessary dependencies (e.g. configuration, utility module, etc.) on the application kernel, and they do so through the level of abstraction. The abstraction layer provides access to the application core, so that the presentation and main layers can be separated.

Among the main templates used in the project, let's focus on: façade (Facade), observer (Observer), factory (Factory), dependency injection (Dependency Injection). The Angular framework used to create the system has its own DI structure, which is usually used in the development of web applications to increase their efficiency and modularity. Dependencies are services or objects required by a class to perform its function. Using the implementation of dependencies, the object simply provides a property that is able to store a link to the desired type of service; And when an object is created, a reference to the implementation of the desired type of service is automatically inserted into this property (field) using the means of the environment. The implementation of dependence is more flexible because it becomes easier to create alternative implementations of this type of service, and then indicate which implementation should be used in, for example, a configuration file, without changes in the objects that use this service.

The factory method is a conciliation pattern, that is, associated with the creation of an object. In the Factory template, we create an object without revealing the creation logic for the client, and the client uses the same common interface to create a new type of object, that was shown in Figure 2.

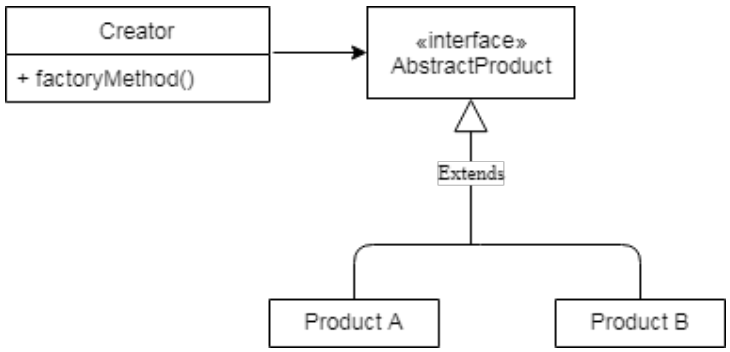

Figure 2: Factory method

The objects involved in this scheme are:

- Creator - the factory object that creates new products sells the "factoryMethod", which returns the newly created products;
- AbstractProduct - declare the interface for products;
- ConcreteProduct - the product that is created, all products implement the same interface (properties and methods),

The main idea is to use a static function (factory static method) that creates and returns instances, hiding the details of the class modules from the user.

The factory method is one of the basic design principles for creating an object, allowing customers to create library objects (explained below) so that it does not have a close relationship with the hierarchy of classes of the library.

When designing complex systems, the principle of decomposition is most often used, in which a complex system is divided into smaller and simpler subsystems. Moreover, the level of decomposition (its depth) is determined exclusively by the designer. Thanks to this approach, individual components of the system can be designed in isolation, then integrated together. However, there is, obviously at first, a problem – the high connectivity of the modules of the system. This is manifested, firstly, in the large amount of information that the modules exchange with each other. In addition, for such communication, some modules must have sufficient information about the nature of other modules. The façade is appropriate to use when it is necessary:

- to simplify access to a complex system;
- to create levels of access to the system;
- to add resistance to changes in subsystems;
- to reduce the number of strong connections between the client and the subsystem but leave access to full functionality.

The reason of using the Observer pattern is that system needs to maintain consistency between related objects without making the classes dependent on each other. For example, in cases where an object should be able to report other objects without making any assumptions about what these objects are. Dynamic relationships can exist between observers and subjects and when using other patterns. However, it is not so easy to implement when the different parts of our application are closely related.

4. Mathematical model CALPUFF

The CALPUFF mathematical model is a multilayer dispersion model, with an integrated Lagrange modeling system that simulates the influence of variable meteorological conditions of time and space to simulate the scattering of atmospheric pollutants. CALPUFF can be applied to a wide range of scenarios, including long-range effects (over 50 km of emissions) that cannot be simulated using traditional flare models (e.g. AERMOD) [18, 19]. This model uses the Gaussian equation to characterize atmospheric processes that disperse the pollutant emitted by the source, based on emissions and meteorological inputs [20, 21]. The concept of this model can be summarized as shown in Figure 3.

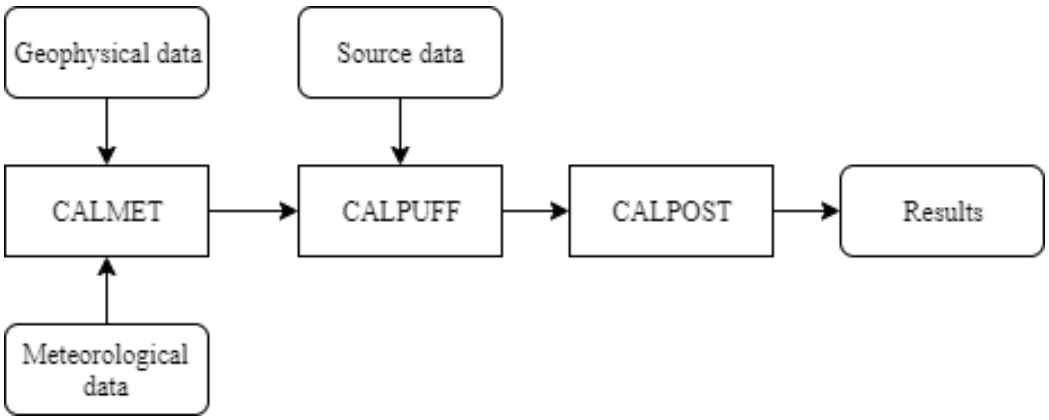

Figure 3: Concentration calculation scheme, based on WRF and CALPUFF

The modeling system consists of three main components and multiple programs for post-processing and preliminary data processing. CALMET – diagnostic three-dimensional meteorological model; CALPUFF – air quality dispersion model; CALPOST – post-processing package.

The equations used in CALPUFF are:

$$C = \frac{Q(s)}{2\pi\sigma y^2} G exp\left[\frac{-R^2(s)}{(2\sigma_y^2(s))}\right], \quad (1)$$

$$G = \frac{2}{(2\pi)^{1/2}\sigma_z} \sum_{n=-\infty}^{\infty} exp\left[\frac{-(H_e+2nh)^2}{(2\sigma_z^2)}\right], \quad (2)$$

where C – is the ground-level concentration at the distance s (м) from the source, Q – the pollution mass in the puff, σ_z – is the standard deviation (m) of the Gaussian distribution in the along-wind direction, σ_y – is the standard deviation (m) of the Gaussian distribution in the cross-wind R – is the distance from the center of the puff to the receptor, s – is the distance travelled by the puff (m), H_e – is the effective height above the ground of the puff center (m), h – is the mixed-layer height (m), G – is vertical term (m) of Gaussian equation [20, 21]. CALMET submodel needs surface and upper air meteorological data [21].

A summary of the required meteorological data is given in Table 1. Table 2 describes the input parameters (the scenario that initiates pollution) for constructing the spread of pollution.

Table 1

Meteorological data required for CALPUFF [20]

Meteorological Data	Geophysical Data	Upper Air Data
Wind Speed	Terrain elevations	Wind Speed
Wind direction	Land use categories	Wind direction
Temperature	Albedo	Temperature
Cloud cover	Bowen ratio	Pressure
Ceiling height	Soil heat flux	Elevation
Surface pressure	Anthropogenic heat flux	
Relative humidity	Vegetative leaf area index	
Precipitation rates		
Precipitation type code		

Data from WRF (Weather Research & Forecasting Model) was taken as meteorological data:

- Domain size 80km x 100km;
- Distribution capacity 2km;
- Meteorological forecast for 72 hours (using GFS (Global Forecast System), 12 hours for SpinUp);

Figure 4 shows the direction and speed of the wind received from WRF.

Table 2

CALPUFF parameters

Name	Description	Value
X Coordinate	easting value of the point source in km units	670.1 km
Y Coordinate	northing value of the point source in km units	799.6 km
Stack Height	the stack height of the point source in meters above ground level (m AGL)	200 m
Base Elevation	the ground elevation at the location of the point source stack in meters above sea level (m ASL)	132 m
Stack Diameter	the inner diameter of the stack in meters	3 m
Exit velocity	the exit velocity at the stack tip in units of m/s	5 m/s
Exit temperature	the temperature of the plume exiting the stack in Kelvin (K) units	355 K
Emission Rates	the rate of constant emissions from the stack; units are defined in the 'emission_units' argument	5.79E07

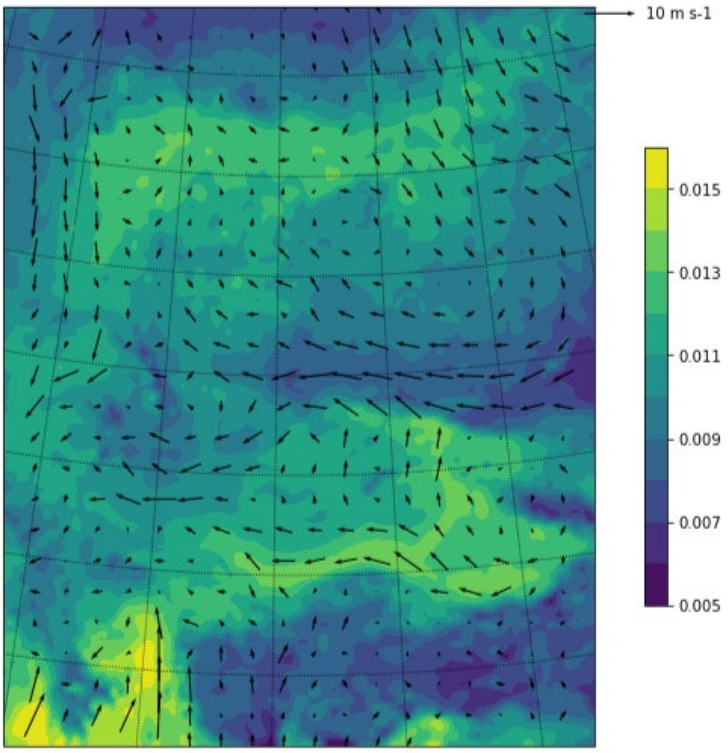

Figure 4: Wind direction and speed obtained using WRF

5. Description of the information system

As a way to transfer data, the architectural approach REST, based on the HTTP protocol, was used and allows you to transfer data in formats other than XML. The system implements a data caching system that provides an opportunity to speed up the issuance of information from the server. The project stores all user behavior, the position of the map, the selected mode of displaying information, etc.

The client part of the information system is SPA (Single Page Application), which provides the following advantages: no need to re-initialize the page every time, data is cached in the browser, data is transferred instead of templates.

The system consists of the following modules:

- page module (used to implement lazy loading);
- common module (implements the fundamental functionality of the framework, including directives, location services used in routing, HTTP services, localization support, etc.);
- modules with functionality (a module containing components, decorators, etc., which perform certain functionality and can be imported into any page module).

In addition, the results of the CALPUFF model were saved as a point cloud (with a minimum time interval of 1 hour) in GeoJSON format on Amazon S3 (Amazon Simple Storage Service) and displayed as a vector layer on the map using the Mapbox GL library. To visualize the presentation of data, user can switch between layers using a time-flow slider, also user can select a pollutant. The system allows user to export the selected layer as a raster image to a local machine.

The system consists of the following pages:

- general map;
- map of wildfires in Ukraine;
- pollution emission map from Arcelormittal (Kryvyi Rih).

On the general map, user can: view the current map of radiation contamination by station (more than 4000 stations) in Figure 5, US AQI (United States Air Quality Index) index calculated by stations (more than 12000 stations) and calculated raster representation of this index (by using interpolation) for Europe, North and South America and some countries of Asia and Oceania in Figure 6.

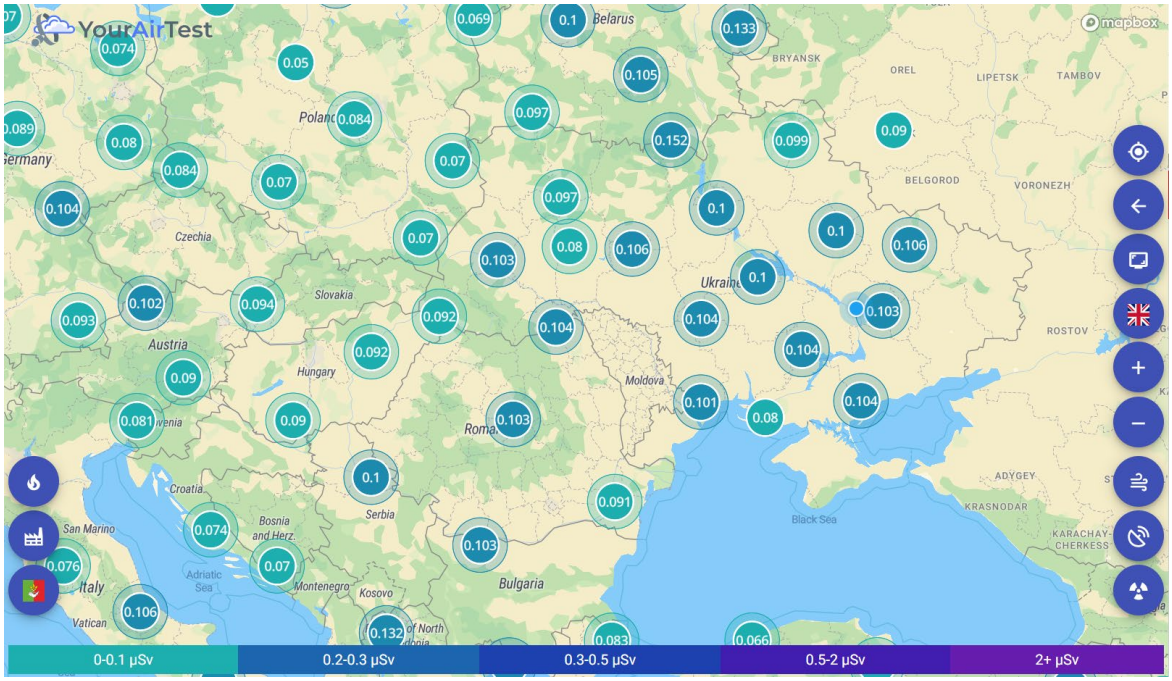

Figure 5: Radiation contamination map

Figure 6: Air quality map calculated by US AQI

Map of fires in Ukraine (current period March-April 2022) on which the user can: view on the map of the fire by the date chosen by him. By selecting fire (user must click fire marker), user open dialog modal and see:

- detailed information on the fire (area of damage, location of the fire, temperature at the point, fire capacity, concentration and amount of emissions of substances into the air)
- simulation of fire spread;
- compare satellite images before, during and after the fire;
- export the data as raster file.

Figures 7 and 8 show map with fires and dialog with information about selected wildfire.

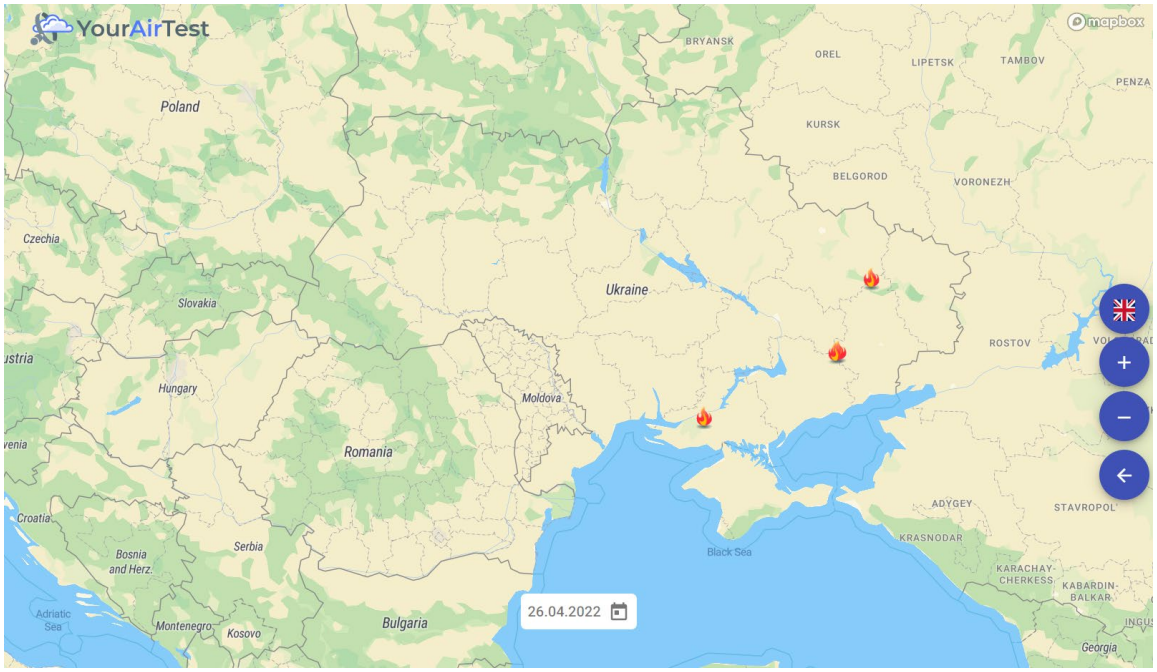

Figure 7: Wildfire map

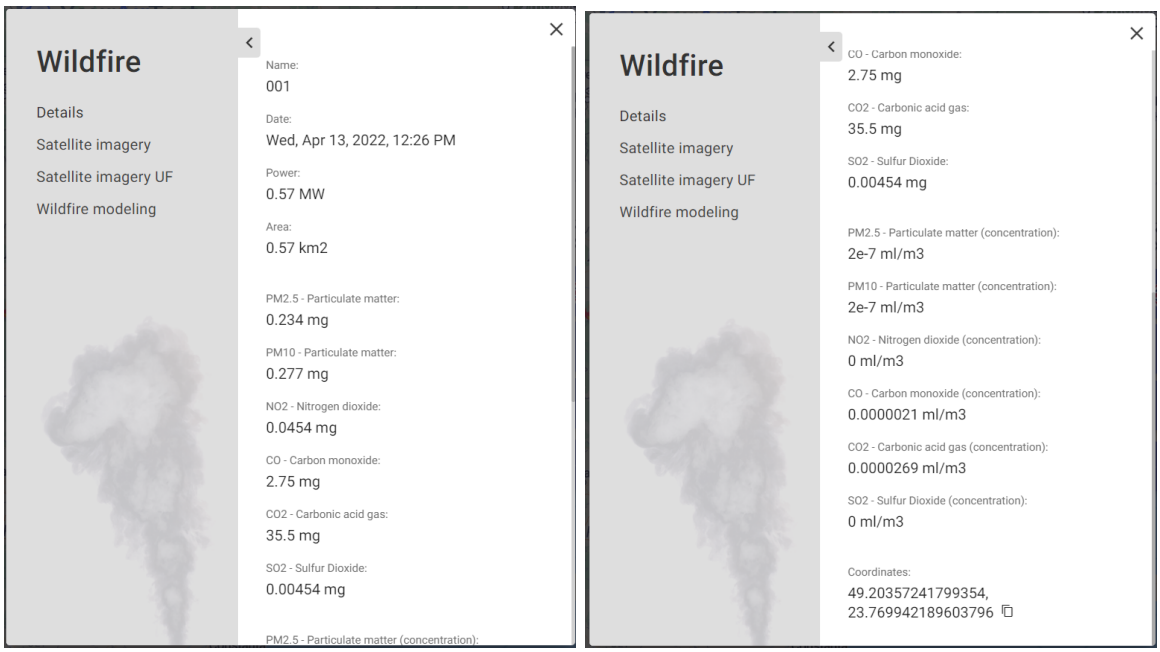

Figure 8: Wildfire dashboard: a) general information and amount of pollution; b) contamination concentration and location

Pollution emission map from Arcelormittal (Kryvyi Rih) visualize results of using the predictive WRF model and multilayer scattering model with integrated Lagrange modeling system CALPUFF. User can view history of pollution emission by using player (2 days, with 1 hour interval, auto play is available), and export selected stage as raster image. This interface is shown in Figure 9.

It is possible to choose a contaminant that is visualized. It can be switched between 4 gases:

- H_2S – hydrogen sulfide, a hazard class 3 toxic gas that acts directly on the nervous system;
- CO – carbon monoxide, affects the cardiovascular system and the central nervous system;
- No_2 – nitrogen dioxide, changes the composition of the blood, reduces hemoglobin content;
- So_2 – sulfur dioxide, leads to cardiovascular and oncological diseases, and also causes irreparable harm to nature.

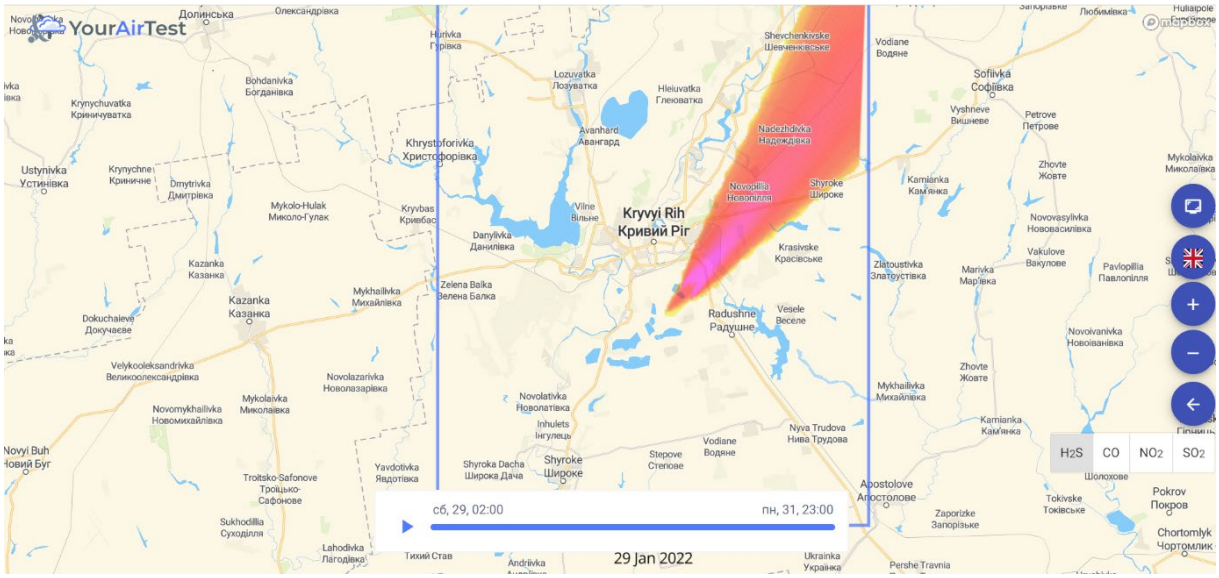

Figure 9: Pollution emission map from Arcelormittal (Kryvyi Rih)

6. Conclusions

The developed architecture for data collection avoids delays in server execution threads by assigning tasks to a separate thread of execution. This system also provides opportunities for quick data exchange. The existence of a large number of auxiliary services is compensated for by containing the project in Docker. The use of REST architecture offers more flexibility in creating the client and server parts, allowing for parallel development and simplifying expansion.

Using CALPUFF model together with the data of the WRF forecast model make possible to visualize the pollution emission in Kryvyi Rih according to the following indicators H_2S, CO, No_2 , So_2. Forecast for 2 days is constructed and visualized on the client part in the form of a vector layer on the map with the possibility of exporting to a local machine.

Factory, Facade, Dependency Injection, Observer patterns were used to simplify client part development of information system. It solves the problem of data exchange in the project, access to services, the problem of directly creating class instances, the problem of a thick client.

Using architectural approaches and technologies described in chapter 2, a system was developed, which:

- aggregates data from ground stations and satellite data;
- builds interactive views of aggregated data;
- provides links or images generated in the system, which can be shared with other users on social networks.

The described architecture is used in the YourAirTest air monitoring information system [18].

7. References

[1] V.V. Hnatushenko, D.K. Mozgovyi, V.V. Vasyliev Satellite Monitoring of Deforestation as a Result of Mining. Naukovyi Visnyk Natsionalnoho Hirnychoho Universytetu, Dnipropetrovsk, 2017. № 5 (161). С. 94-99.

[2] World Health organization, Air pollution (who.int), 2023 URL: https://www.who.int/health-topics/air-pollution - tab=tab_1

[3] V. Hnatushenko, P. Kogut, M. Uvarov. On Flexible Co-Registration of Optical and SAR Satellite Images. In: Babichev S., Lytvynenko V., Wójcik W., Vyshemyrskaya S. (eds) Lecture Notes in Computational Intelligence and Decision Making. ISDMCI 2020. Advances in Intelligent Systems and Computing, vol 1246. Pp. 515-534. Springer, Cham. doi: 10.1007/978-3-030-54215-3_33.

[4] V.J. Kashtan, V.V. Hnatushenko, Y.I. Shedlovska. Processing technology of multispectral remote sensing images. 2017 IEEE International Young Scientists Forum on Applied Physics and Engineering (YSF), 2017. doi:10.1109/ysf.2017.8126647.
[5] M. Usama. Urban Air Quality Measurements: A Survey. Preprints.org; 2022. doi: 10.20944/preprints202204.0232.v1.
[6] S. Abu El-Magd, G. Soliman, M. Morsy, et al. Environmental hazard assessment and monitoring for air pollution using machine learning and remote sensing. Int. J. Environ. Sci. Technol. (2022). doi: 10.1007/s13762-022-04367-6.
[7] Andrew Rowley, Oktay Karakuş. Measuring Air Quality via Multimodal AI and Satellite Imagery. arXiv. 2022. doi: 10.48550/arXiv.2211.00780.
[8] L. Chen, J. Wang, H. Wang, T. Jin. Urban Air Quality Assessment by Fusing Spatial and Temporal Data from Multiple Study Sources Using Refined Estimation Methods. ISPRS Int. J. Geo-Inf. 2022, 11, 330. doi: 10.3390/ijgi11060330.
[9] T. Singh, N. Sharma, Satakshi, M. Kumar. Analysis and forecasting of air quality index based on satellite data. Inhal Toxicol. 2023 Jan-Feb;35(1-2):24-39. Epub 2023 Jan 5. PMID: 36602767. doi:10.1080/08958378.2022.2164388.
[10] Ridzuan, Nurfairunnajiha, Ujang, Uznir, Azri, Suhaibah, Choon, Tan. (2020). Visualising urban air quality using aermod, calpuff and CFD models: A critical review. Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLIV-4/W3-2020, 355–363, doi:10.5194/isprs-archives-XLIV-4-W3-2020-355-2020, 2020.
[11] Boadh, Rahul, Rajoria, Yogendra, Prasad, V.Ramachandra. (2022). Air Pollution Dispersion Using Coupled AERMOD-WRF Modeling System and Generation of Gridded Emission Inventory of NO_X over Nagpur. doi:10.5772/intechopen.108230.
[12] Fadhil, Nyaz, Jaf, Saba. (2022). Air Desperation Quality by Using AERMOD Software Program. Malaysian Journal of Computer Science. 7. 2022-2023. doi:10.5281/gsjcs.2022.7098203.
[13] Shubbar, Ramiz & Lee, D. & Gzar, Hatem & Rood, Arthur. (2019). Modeling Air Dispersion of Pollutants Emitted from the Daura Oil Refinery, Baghdad- Iraq using the CALPUFF Modeling System. Journal of Environmental Informatics Letters. 10.3808/jeil.201900014.
[14] Richard Bullington-McGuire. (2020). Docker for Developers: Develop and run your application with Docker containers using DevOps tools for continuous delivery.
[15] Ćatović, Amar & Buzađija, Nevzudin & Lemeš, Samir. (2022). Microservice development using RabbitMQ message broker. Science, Engineering and Technology. 2. 30-37. doi:10.54327/set2022/v2.i1.19.
[16] Docker & Kubernetes 3: minikube Django with Redis and Celery - 2020 URL: https://bogotobogo.com/DevOps/Docker/Docker_Kubernetes_Minikube_3_Django_with_Redis_Celery.php
[17] R. Baida, M. Andriienko, M. Plechawska-Wójcik (2020). Performance analysis of frameworks Angular and Vue.js. Journal of Computer Sciences Institute, 14, 59-64. doi:10.35784/jcsi.1577
[18] Your Air Test // URL: https://youairitest.com
[19] R. Zhang, M. Li, H. Ma (2022). Comparative study on numerical simulation based on CALPUFF and wind tunnel simulation of hazardous chemical leakage accidents. Front. Environ. Sci. 10:1025027. doi: 10.3389/fenvs.2022.1025027.
[20] Pet. Pakchotanon, (2022). Atmospheric Dispersion of Gaseous Amine Emitted from Absorption-Based Carbon Capture Plants in Saskatchewan, Canada. Energies. 15. doi:10.3390/en15031221.
[21] Y. Bezyk, D. Oshurok, M. Dorodnikov, I. Sówka. (2020). Evaluation of the CALPUFF model performance for the estimation of the urban ecosystem CO2 flux. doi:10.1016/j.apr.2020.12.013.