



SHANGHAI MARITIME UNIVERSITY

Report of National Air Pollution Visualization System

|  |  |
| --- | --- |
| **Group Number：** | **Group I of Software Project** |
|  | **Management** |
| **Project Name：** | **Report of National Air Pollution** |
|  | **Visualization System** |
| **Subject Major：** | **Computer Technology** |
| **Team Numbers：** | **Wenjie Xu 202230310220** |
|  | **Qianlong Lu 202230310293** |
|  | **Ruirui Zhao202230310197** |
| **Instructor：** | **Jin Liu** |
| **Finish Date:** | **May 2023** |

Catalog

[Chapter1 Project Introduction 2](#_Toc136270305)

[1.1 Project Background 2](#_Toc136270306)

[1.2 System Overview 4](#_Toc136270307)

[Chapter2 Design 4](#_Toc136270308)

[2.1 Detailed Design 4](#_Toc136270309)

[2.2 Detailed Design Specification 5](#_Toc136270310)

[2.2.1 Back-end module design notes 5](#_Toc136270311)

[2.2.2 front-end module design description 9](#_Toc136270312)

[2.2.3 Database design 15](#_Toc136270313)

[2.3 Technology Introduction 15](#_Toc136270314)

[2.3.1 Introduction of vue3 15](#_Toc136270315)

[2.3.2 Introduction of echart 17](#_Toc136270316)

[2.3.3 Introduction of SpringBoot 18](#_Toc136270317)

[Chapter3 Schedule 19](#_Toc136270318)

[3.1 Arrangement 19](#_Toc136270319)

[3.2 Team Meeting 19](#_Toc136270320)

[3.3 Initial Risk Plan 20](#_Toc136270321)

[3.4 Milestone 20](#_Toc136270322)

[3.5 Plan of Air Quality Visualization System Implementation 23](#_Toc136270323)

[Chapter4 Final Product Demo 23](#_Toc136270324)

[Chapter5 Project Execution Detail 25](#_Toc136270325)

[Chapter6 Risk Plan 26](#_Toc136270326)

[6.1 Introduction 26](#_Toc136270327)

[6.2 Project Risk Management Organization 27](#_Toc136270328)

[6.3 Project Definition Risk Management Form 27](#_Toc136270329)

[6.3.1 Project Risk Level Definition 27](#_Toc136270330)

[6.3.2 Project Risk Probability and Impact Definition 28](#_Toc136270331)

[6.3.3 Project Risk Status Definition 28](#_Toc136270332)

[6.4 Project Risk Management Schedule 29](#_Toc136270333)

[Chapter7 Test Plan & Test Cases 30](#_Toc136270334)

[7.1 Test Plan 30](#_Toc136270335)

[7.1.1 schedule 30](#_Toc136270336)

[7.1.2 Design Notes 32](#_Toc136270337)

[7.2 Test Case 35](#_Toc136270338)

[7.2.1 Junit Introduction 35](#_Toc136270339)

[7.2.2 The way JUnit works 36](#_Toc136270340)

[7.2.3 JUnit Benefits 37](#_Toc136270341)

[7.2.4 Our JUnit Cases 38](#_Toc136270342)

[Chapter8 Lesson Learnt 40](#_Toc136270343)

[Chapter9 Conclusion 40](#_Toc136270344)

[Chapter10 Future Plan For Next Cycle 41](#_Toc136270345)

# Chapter1 Project Introduction

## 1.1 Project Background

In recent years, air quality monitoring stations have collected a large amount of air quality data with high-dimensional and time-series characteristics, and it is challenging to use such data to analyze and understand air pollution transmission patterns and provide effective suggestions for decision makers.

The prevention and control of air pollution in China has started since 1970, but it was not until 2013 when the "Ten Articles of the Atmosphere" were introduced that air pollution control entered a rapid development stage.

Since 2013, China's air pollution control has achieved great results in many aspects, and the air quality

The air quality has improved significantly, and the total emissions of major air pollutants have dropped significantly. However, with the "Blue Sky Defense War" in 2018, air quality has entered an assertive phase.

The pain points, difficulties and shortcomings of air governance are gradually revealed. In terms of atmospheric quality, PM2.5 has not yet reached the standard, and NO2 concentration is decreasing slowly.

In terms of air quality, PM2.5 has not yet reached the standard, NO2 concentration is decreasing slowly, O3 is increasing instead of decreasing, and it is still difficult to improve air quality significantly. Therefore, in the current attack

Therefore, in the current period, the analysis and investigation of the spatial and temporal distribution pattern of air pollution and its evolution can provide reference for the formulation of long-term effective prevention and control measures.

With this background, this system mainly carries out the following work:

(1) For the existing large-scale data, visualization is used to visually and effectively display the air pollutants and air quality in each province and city from 2013 to 2018.

To visualize the distribution patterns and evolution of each pollutant and air quality in each province and city from 2013 to 2018;

Perform month-by-month sliding PCC correlation calculations on existing data to visually analyze the correlation within pollutants and between pollutants and meteorological factors.

(2) Visual analysis of the correlation within pollutants and between pollutants and meteorological factors;

(3) Analysis of air pollution fluctuation patterns to enable monitoring of important time points where anomalies or significant changes occur;

(4) Regional clustering of pollutant trends per sliding quarter, and spatial and temporal comparison analysis to obtain spatial and temporal distribution patterns of air pollution.

spatial and temporal distribution patterns of air pollution through spatial and temporal comparison analysis;

(5) Based on the results of visual analysis and assessment, the spatial and temporal distribution patterns and spatial and temporal evolution of air pollution over a six-year period are summarized.

(6) Based on the results of visual analysis and assessment, the spatial and temporal distribution patterns and spatial and temporal evolution of air pollution over six years were summarized.

## 1.2 System Overview

Through various graphical visualizations and interactions, we analyze the impact of six air pollutants and three meteorological factors on air quality in different provinces and cities in China over the past six years from different perspectives. The graphs and maps also help us to discover the characteristics and causes of pollutant concentrations from them.

# Chapter2 Design

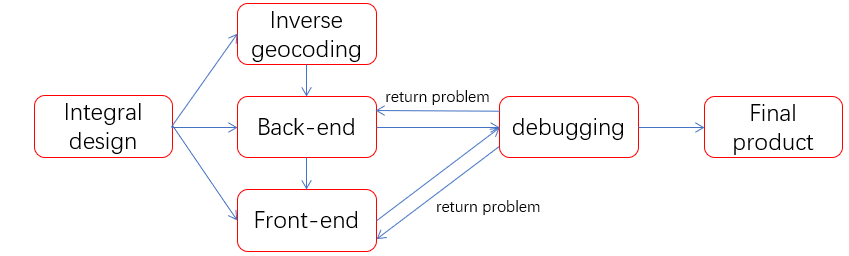
## 2.1 Detailed Design

The visualization system in our group is divided into two parts, namely the front end and the back end.

The front-end will use Vue3.0 and Echarts to visualize the data, and the data used for the display will be obtained by requesting the back-end interface.

Before coding the back-end interface we will reverse geocode and integrate the data set provided by the data visualization contest and then divide it into 6 tables by year into MySQL. After that we will code the back-end interface using the Spring Boot framework.

The initial entire page will contain five modules that present China's six year pollutant data at different granularity. By drilling down to provinces and cities, the pollutant situation in different regions is presented in more detail.



As shown in this figure, our project process includes the following parts: After completing the design of the entire system, we officially engage in development work. The primary task is to process the longitude and latitude information in the database through inverse geocoding to obtain a mapping of longitude and latitude to an accurate geographical location. After completing this work, the back-end can start development work. The back-end continuously provides data interfaces, and the front-end uses the interface for development. Each function completed, the tester tests its reliability, and if there are any issues, they can promptly feedback to the front-end and back-end. This is a pipeline working mode. In this cycle, we continuously iterate over our products and finally complete the development goals.

## Detailed Design Specification

### 2.2.1 Back-end module design notes

#### 2.2.1.1China map of the average pollution interface by province -getAllProvincePollutions

**(1)Module description**

Get the six-year average of each pollutant for each province of the China map.

**(2)Input**

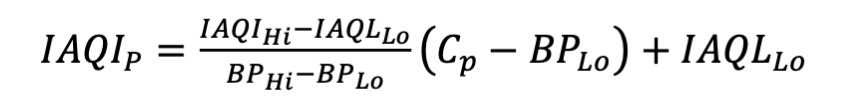
None

**(3)Output**

A list of six-year average values of each pollutant for each province is available.

**(4)Algorithm design**

The first step is to calculate the AQI values. The first step is to cross-reference the classification of each pollutant against the the first step is to calculate the air quality index (IAQI) based on the measured concentrations of fine particulate matter (PM2.5), respirable particulate matter (PM10), sulfur dioxide (SO2), nitrogen dioxide (NO2), carbon monoxide (CO), ozone (O3), and other pollutants, respectively, according to the following calculation formula:



Then take the maximum value of each pollutant air quality sub-index is the air quality number (AQI for short).

Finally, the 6-year average of pollutants in each province is loaded into Pollution object, and finally the map with the mapping relationship between provinces and Pollution object is loaded into list and returned to the front-end.

#### 2.2.1.2 Obtain the average value of pollutants in each province according to the year -getInfoByYear

**(1)Module description**

Output the average value of pollutants for each province according to the input year.

**(2)Input**

Year

**(3)Output**

Pollutant averages for each province in the year.

#### 2.2.1.3 Interface to obtain urban pollution data according to the name and year of the province -getCityInfoByProvince

**(1)Module description**

Obtain urban pollution data by province name and year.

**(2)Input**

Province、Year

**(3)Output**

Provincial pollutant data in a given year.

#### 2.2.1.4 Obtain the average provincial pollutants in the last six years - getSomeAvgCount

**(1)Module description**

Get six-year national average of all provinces by pollutant.

**(2)Input**

None

**(3)Output**

Six-year national average of all provinces for each pollutant.

#### Interface for obtaining average urban pollutant values for the target year - getSomeCityAvgCount

**(1)Module description**

Obtain the pressure, temperature and humidity of a city for the last six years.

**(2)Input**

City

**(3)Output**

The pressure, temperature and humidity of the city for the last six years.

#### 2.2.1.6 Obtain pollutant data for the top ten provinces in the country for the last six years - getTenProvinceAsc

**(1) Module description**

Get the top 10 AQI provinces each year, with other pollutants returned in order of AQI.

**(2) Input**

None

**(3) Output**

Top 10 AQI provinces per year and the value of each pollutant in that province.

#### 2.2.1.7 Interface to obtain the national average of pollutants for the last six years -getSixAverage

**(1) Module description**

Get six-year national pollutant averages.

**(2) Input**

None

**(3) Output**

Six-year average for each pollutant.

#### Obtain the pollutant data of the top ten cities in a province in the last six years -getTenCityAsc

1. **Module description**

Get the top 10 city pollutant data for each year in six years for the specified provinces.

**(2) Input**

Province

**(3) Output**

Get the top ten city pollutant data for the specified province for each of the six years, along with their pollutant values.

#### 2.2.1.9 Get Pollutant Average Data Interface - Get the average pollutant data of a province in the last six years -getSixAverageByProvince

**(1) Module description**Obtain the average pollutant data of a province for the last six years

**(2) Input**

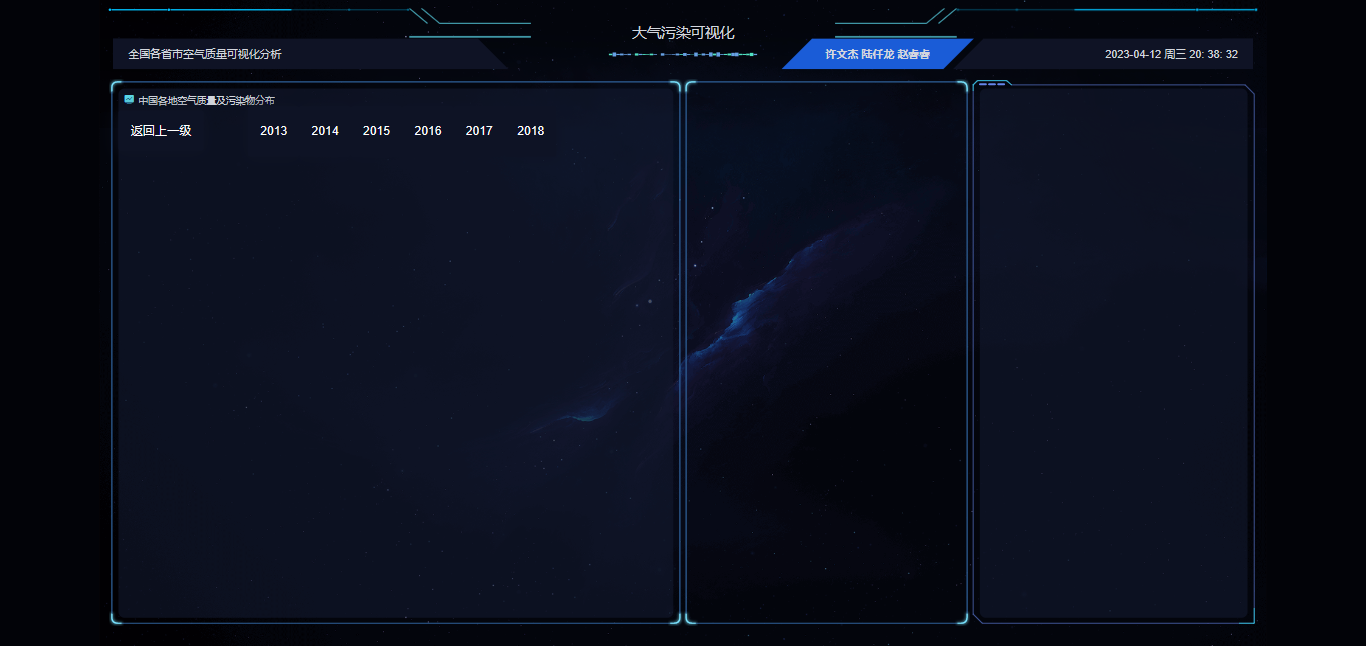
Province

**(3) Output**

The six-year average of each pollutant in the province.

### 2.2.2 front-end module design description

#### 2.2.2.1 index.vue

****

Title (Text): Visualization of Air Pollution

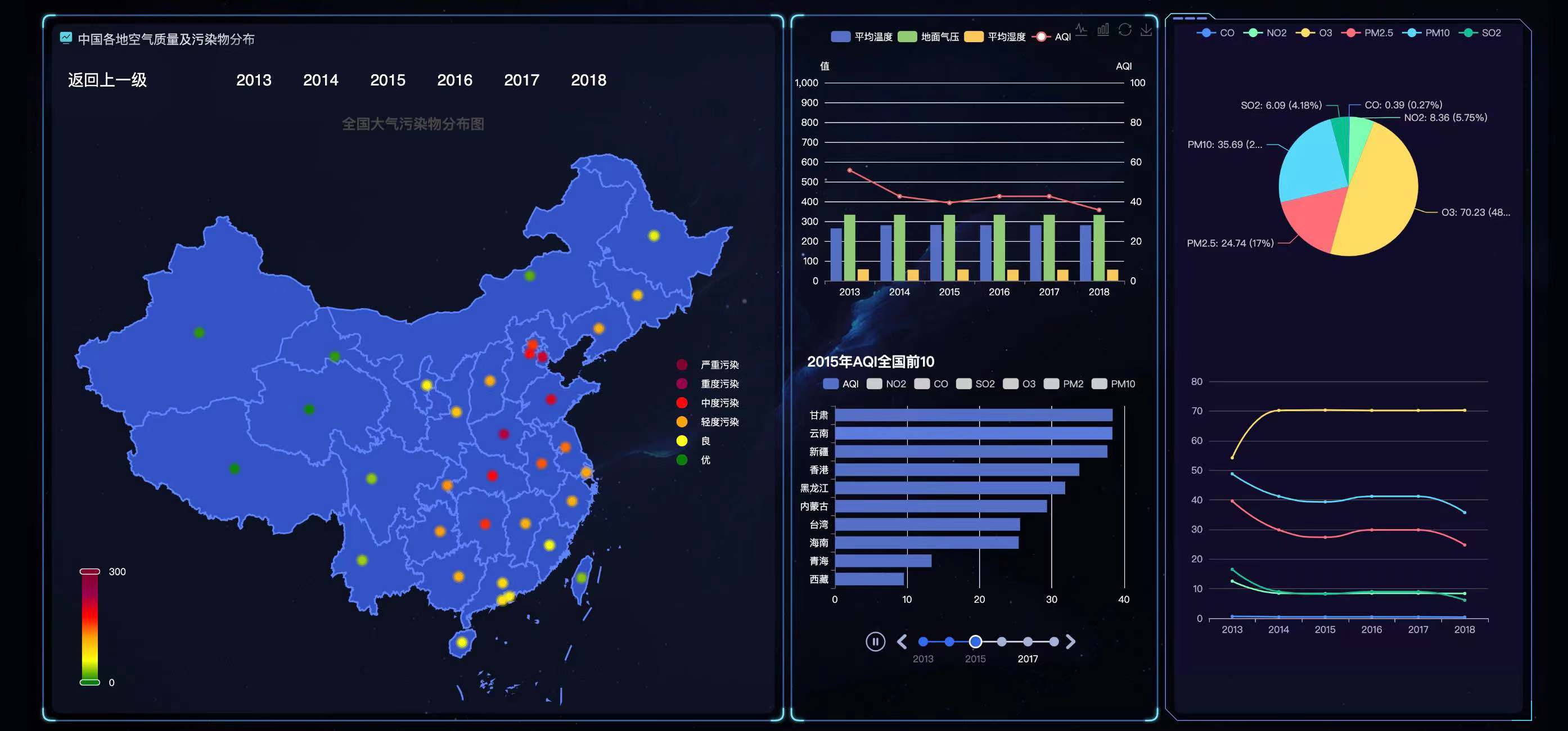
Subtitle (text): Visualization of air quality in provinces and cities across the country

Names of team members (text)

Current date and time (text)

Four modules: left1, center1, center2, and right correspond to left, upper center, lower center, and right, respectively

#### 2.2.2.2 Realistic national and provincial maps and information -left1.vue



Title (Text):Air quality and pollutant distribution across China

Return to previous level (button):Click to call returnLevel to realize the provincial map drill back to China map

2013~2018 (button): click on 20XX to call getInfoByYear20XX() to view the number of air pollution information of different years

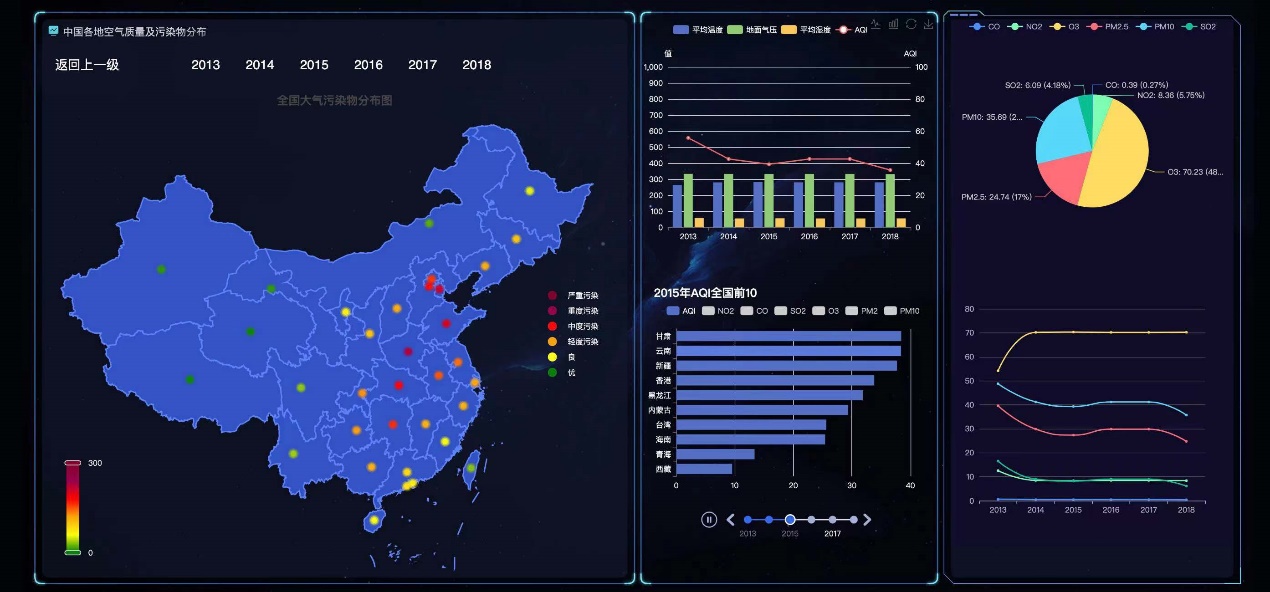
Map: call drawChina to draw a map, register the available maps through echart.registerMap(), set five colors to represent "severe pollution", "heavy pollution", "moderate pollution", "moderate pollution", and "high pollution", "moderate pollution", "light pollution", "good" and "excellent", and set five labels in each province of the map. The location of the provincial capital shows light dots of corresponding colors according to the local air quality condition. By selecting the range on the color spectrum, you can display only the light dots of certain provinces, and mouse over a certain identity will show the specific information of that province.

Province: When clicking on a province, call getDown() to drill down, all pages will display the air pollution data information of this province, when clicking on provinces as Shanghai, Beijing, Chongqing, Tianjin, Hong Kong SAR, Macau SAR and Taiwan Province, no drill down operation will be performed, after the drill down, data information of cities in the province will be displayed. The following is the effect of drilling down after clicking on Inner Mongolia Autonomous Region:





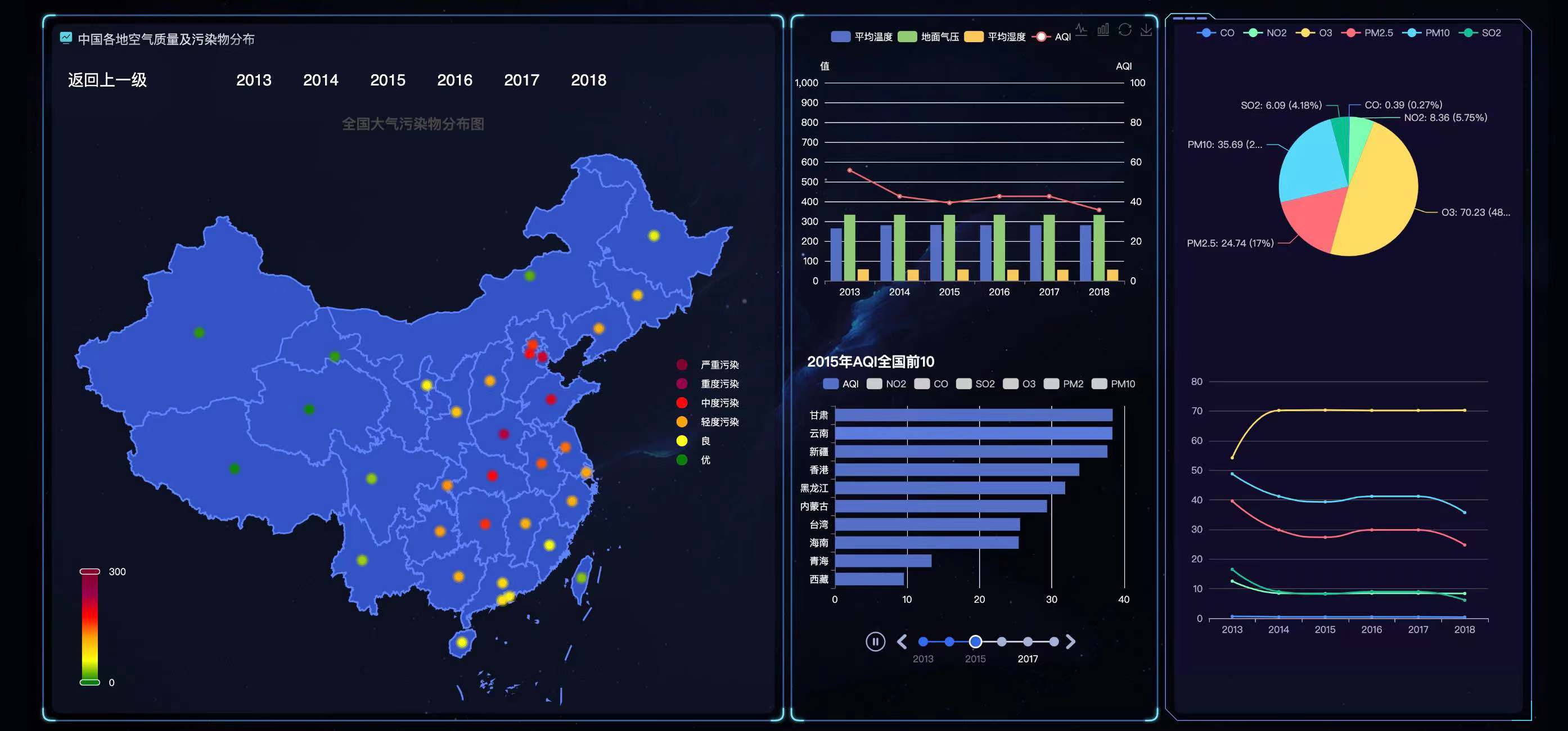
#### 2.2.2.3 Relationship between AQI and mean temperature, ground pressure, and mean humidity over the six-year period -center1.vue



Air quality analysis bar chart: the horizontal coordinate is 2013~2018, the vertical coordinate is the value, call getData to get the data information, display the four fold lines of average temperature, ground pressure, average humidity, and AQI.

Average temperature, ground pressure, average humidity, AQI (button): you can choose to hide or show the fold after clicking.

#### 2.2.2.4 Top 10 provinces (cities) with the smallest AQI values nationwide according to AQI order 2013-2018 -center2.vue



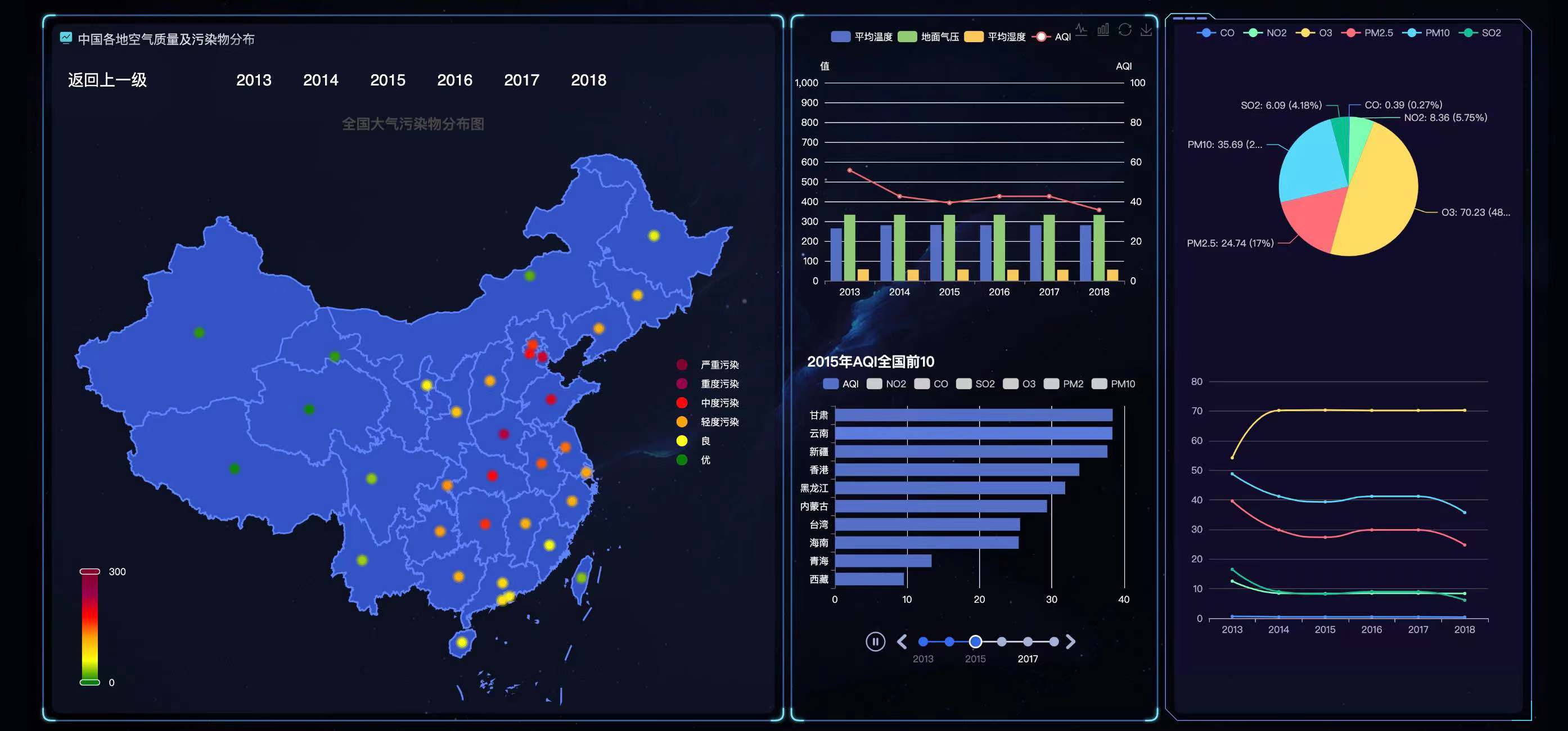
Title: 20XX AQI National Top 10

Timeline: 2013~2018, each year corresponds to a bar chart

AQI ranking bar chart: the vertical coordinate is the name of the top ten provinces and cities, the horizontal coordinate is the value, call getAqiByAsc() in getData to get the data and sort them, dataFormatterNO2(), dataFormatterCO(), dataFormatterSO2(), dataFormatterPM2(), dataFormatterPM10() to get the data of NO2, CO, SO2, O3, PM2.5 and PM10 respectively, dataFormatterO3(), dataFormatterPM2(), dataFormatterPM10() to get the data of NO2, CO, SO2, O3, PM2.5 and PM10 respectively.

CO, NO2, O3, PM2.5, PM10, SO2 (button): you can choose to hide or show this histogram after clicking

#### 2.2.2.5 Trends in the mean values and percentages of pollutant components over 6 years -right1.vue



Pollutant average line graph: the horizontal coordinate is the year, the vertical coordinate is the value, get data by getData, display the average value of CO, NO2, O3, PM2.5, PM10, SO2 for the whole country or a province, where the six lines and the pie chart are combined into one component, when the year changes, the pie chart will change together with the line graph.

CO, NO2, O3, PM2.5, PM10, SO2 (button): you can choose to hide or show the information of this component when you click on it.

Pollutant share sector chart: Show the information of pollutant component share for China or a province.

### 2.2.3 Database design

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Length** | **Null** | **PK** | **Description** |
| 1 | id | int |  | No | Yes | id |
| 2 | pm2 | decimal | 20 | No | No | PM2.5 |
| 3 | pm10 | decimal | 20 | No | No | PM10 |
| 4 | so2 | decimal | 20 | No | No | SO2 |
| 5 | no2 | decimal | 20 | No | No | NO2 |
| 6 | co | decimal | 20 | No | No | CO |
| 7 | o3 | decimal | 20 | No | No | O3 |
| 10 | temp | decimal | 20 | No | No | 温度 |
| 11 | rh | decimal | 20 | No | No | 相对湿度 |
| 12 | psfc | decimal | 20 | No | No | 地面气压 |
| 13 | lat | decimal | 20 | No | No | 中心纬度 |
| 14 | lon | decimal | 20 | No | No | 中心经度 |
| 15 | aqi | decimal | 20 | Yes | No | 空气质量指数 |
| 16 | province | varchar | 255 | Yes | No | 省 |
| 17 | city | varchar | 255 | Yes | No | 市 |
| 18 | monnth | int |  | No | No | 月份 |

## 2.3 Technology Introduction

### 2.3.1 Introduction of vue3

Vue.js 3.0 is a brand new version of the Vue.js framework that brings many new features and changes. The following are some of the major framework changes in Vue.js 3.0:

1. New responsive system

The responsive system in Vue.js 3.0 has been completely upgraded and now uses Proxy proxy objects instead of Object.defineProperty. this makes the responsive system in Vue.js 3.0 more efficient and flexible, as well as easier to debug and maintain.

2. Better componentization

Vue.js 3.0 has also seen significant improvements in componentization. Components in Vue.js 3.0 can now be defined using templates, JSX, or rendering functions. This makes Vue.js 3.0 more flexible and can be better adapted to different development scenarios.

3. Better routing support

Vue.js 3.0 also has improved support for routing. Routing in Vue.js 3.0 can now be implemented using Vue Router 4.0, which makes the configuration of routes much clearer and more concise.

4. Better state management

Vue.js 3.0 provides a new state management library, Vuex 4.0, which uses the Proxy proxy object in a responsive system to implement state management. This makes Vuex 4.0 more efficient and flexible, as well as easier to debug and maintain.

5. Better TypeScript support

TypeScript support in Vue.js 3.0 has also been significantly improved. The type definition file in Vue.js 3.0 is now more complete and accurate, making it easier and more intuitive to use Vue.js in TypeScript.

Summary

Vue.js 3.0 is a completely new version of the Vue.js framework that brings many new features and changes. vue.js 3.0 has a fully upgraded responsive system and provides better componentization, routing support, state management, and TypeScript support. These changes make Vue.js 3.0 more efficient, flexible and easy to maintain.

### 2.3.2 Introduction of echart

ECharts, a pure Javascript charting library, runs smoothly on PC and mobile devices, compatible with most current browsers (IE8/9/10/11, Chrome, Firefox, Safari, etc.), and relies on the underlying lightweight Canvas library ZRender to provide intuitive, vivid, interactive, and highly highly customizable data visualization charts.

#### Rich Chart Types

ECharts provides conventional line, bar, scatter, pie and k-line charts, box charts for statistics, maps, heat maps and line charts for geographic data visualization, relationship charts, treemap for data relationship visualization, parallel coordinates for multidimensional data visualization, funnel charts and dashboards for BI, and supports mixing and matching between charts.

#### Dynamic Data

ECharts is driven by data, and changes in data drive changes in chart presentation. Therefore, the implementation of dynamic data becomes very simple, just get the data, fill in the data, ECharts will find the difference between the two sets of data and then show the change of data through appropriate animation.

#### Mobile Optimization

ECharts and ZRender code refactorings have reduced the size of the core part of the charting library, which needs to be as small as possible for mobile, where traffic is precious.

#### Multidimensional data support and rich visual coding tools

ECharts 3 begins to enhance its support for multidimensional data. In addition to the inclusion of common multidimensional data visualization tools such as parallel coordinates, incoming data can be of multiple dimensions for traditional scatter plots. The visual mapping component visualMap provides rich visual coding to map data of different dimensions to different visual channels such as color, size, transparency, lightness and darkness.

### 2.3.3 Introduction of SpringBoot

(1) Spring Boot is a new framework provided by the Pivotal team and is designed to simplify the initial build and development process of new Spring applications. The framework uses a specific approach to configuration so that developers no longer need to define sample configurations. Traditional spring projects need to use a lot of xml configuration, and complex bean dependencies, with the release of spring3.0 spring team began to use a lot of conventions than configuration ideas to simplify the configuration.

(2) Spring Boot is such a development framework, Spring Boot does not directly complete the work of Spring. To be precise, Spring Boot is just a middleware that integrates Spring features for rapid development of Spring applications, and also provides default configuration for common third-party libraries, so you can almost achieve the purpose of 0 configuration development.

(3) Spring Boot is designed to get up and running as fast as possible with minimal Spring configuration and has a unique approach to building production ready applications. As you can see from the official introduction, the core idea of Spring Boot is "Convention Over Configuration", which is still essentially based on Spring. For those who know Spring or have used it, Spring's cumbersome configuration dazzles many programmers (various XML, Annotation configurations, etc.), and many times it's difficult to quickly locate the error even when it occurs. The Spring Boot framework provides us with default configurations so that developers no longer need to define sample configurations.

# Chapter3 Schedule

## 3.1 Arrangement

Every member of our group all have their own task.Xu Wenjie is responsible for demand analysis,outline design,detailed design and back-end implementation;Lu Qianlong is responsible for main front-end implementation, Zhao Ruirui provide front-end development ideals and requirements and do interface debugging.

## 3.2 Team Meeting

Our team meeting consist of online and offline meeting,we will exchange our progress and problems after the weekly group meeting organized by the tutor, and communicate with each other through WeChat when we have any question.

## 3.3 Initial Risk Plan

And we also will face with some risk,there list just two:

1. The risk of limitation on number of Amap API requests,so we need to find a Chinese geographic,
2. Insufficient understanding of front-end technology,during the process of completing our project,I will learn the knowledge of vue3 framework.There may be other issues that have’t take into account,but we will try our best to solve them.

## 3.4 Milestone

The number of days for each subtask in Milestone is planned according to a 7-hour work week per person. The schedule for each task is shown in the "Project Implementation Plan" Excel file.



**M0（2023.3.23）:**

Wenjie Xu:

1. complete the requirements analysis to determine all the functions to be achieved by the system

2. complete the outline design of the system and determine the overall framework used for system writing

Qianlong Lu, Ruirui Zhao:

1. learn the use of Vue3

M0 Criteria:

1. determine the overall system functionality and technical framework by March 23rd

2. learn the basic use of Vue3 and be able to achieve front and back-end interaction.

**M1（2023.3.30）:**

Wenjie Xu, Qianlong Lu, Ruirui Zhao:

1. complete the Gantt chart of the project plan

2. Complete the first presentation PPT

M1 Criteria:

1. Submit the Gantt chart of project implementation plan to Github

2. Submit the PPT for the first presentation to Github

**M2（2023.4.27）：**

Wenjie Xu:

1. Complete data pre-processing to convert latitude and longitude in the 6-year dataset to provincial and municipal information

2. complete the writing of the first 5 back-end interfaces in the project implementation plan

Ruirui Zhao:

1. test the front-end and back-end interfaces together and find out if there are bugs

Qianlong Lu:

1. finish writing 5 front-end charts

M2 Criteria:

1. convert the latitude and longitude coordinates of all data sets for six years into provinces and cities and store them in the database

2. Submit the code of the first 5 interfaces in the project implementation schedule to Github

3. submit 5 front-end visualization charts to Github

4. test the front-end and back-end data interaction through the first 5 functions, and require 0 bugs

**Beta（2023.5.31）：**

Wenjie Xu:

1. finish writing the last 4 back-end interfaces in the project implementation schedule

Qianlong Lu:

1. finish writing the last 3 diagram codes for the front-end

Ruirui Zhao:

1. test all functional modules

Wenjie Xu, Qianlong Lu, Ruiyuan Zhao:

1. finish the PPT for the final presentation

Beta Criteria.

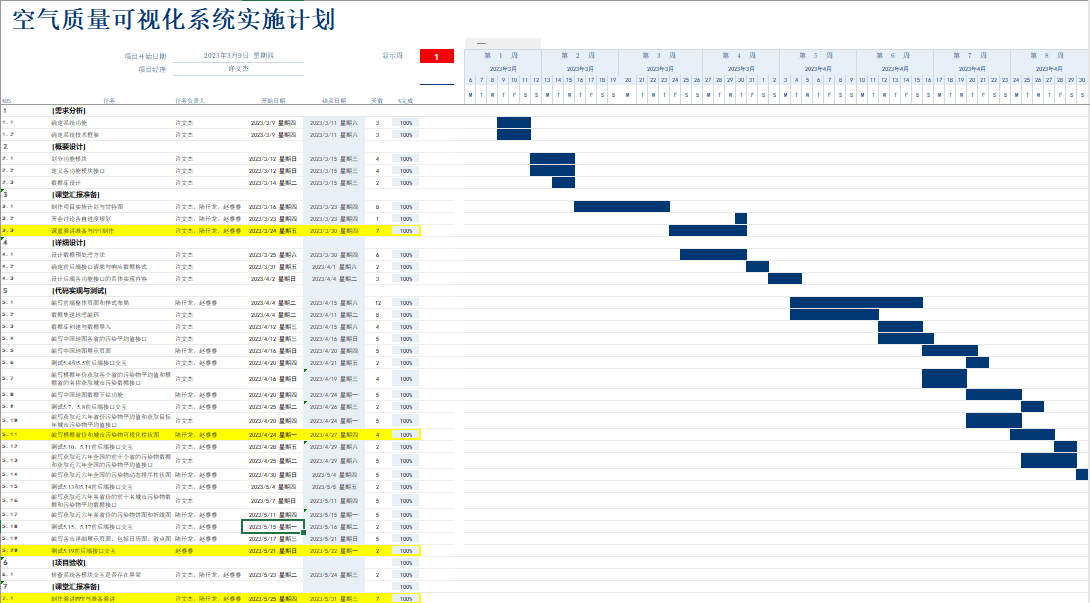
1. Submit the code of the last 4 interfaces in the project implementation plan to Github

2. Submit the code of the last 3 front-end diagrams in the project implementation plan to Github

3. Test the front-end and back-end interactions of all functional modules with 0 bugs

4. Submit the presentation PPT to Github

## 3.5 Plan of Air Quality Visualization System Implementation

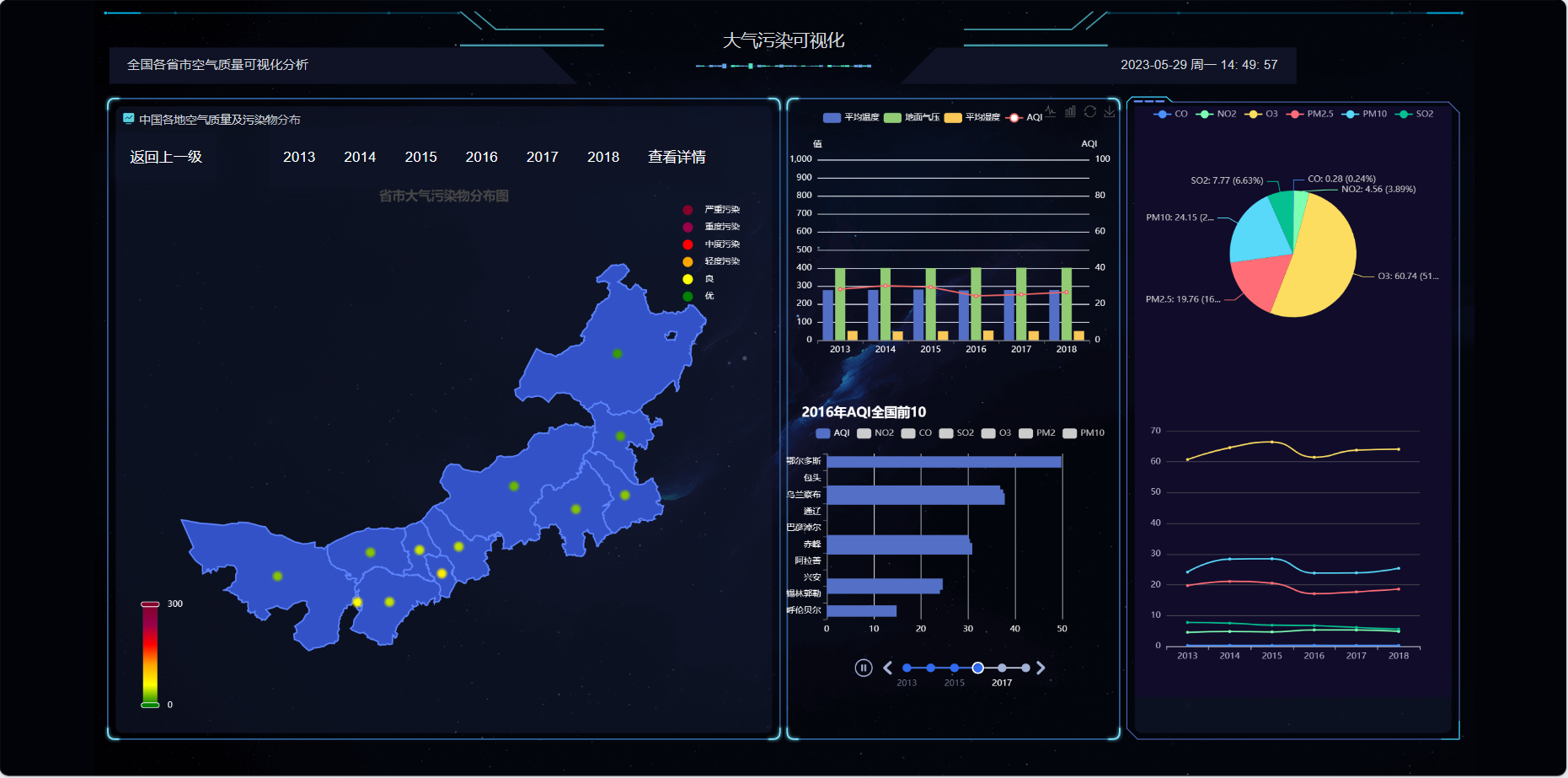


# Chapter4 Final Product Demo



This is our home page, which mainly includes the following content: from left to right, there is a map of China, the trend of average humidity, temperature and other indicators, the ranking of pollutants in different provinces and years, and other related displays.

You can see that there are many control buttons on the page that can display different graphs.

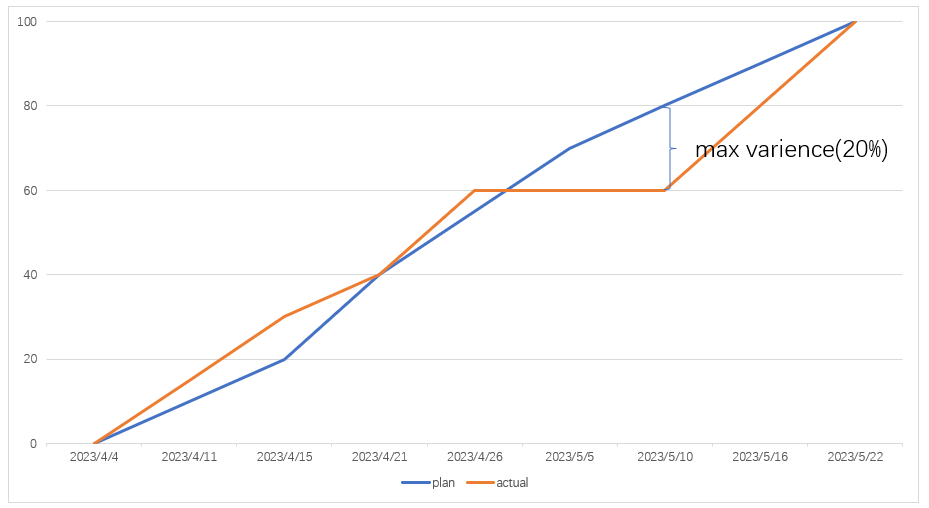


On the left map, you can drill down on any province, and the relevant visualization data on the right will also drill down accordingly.

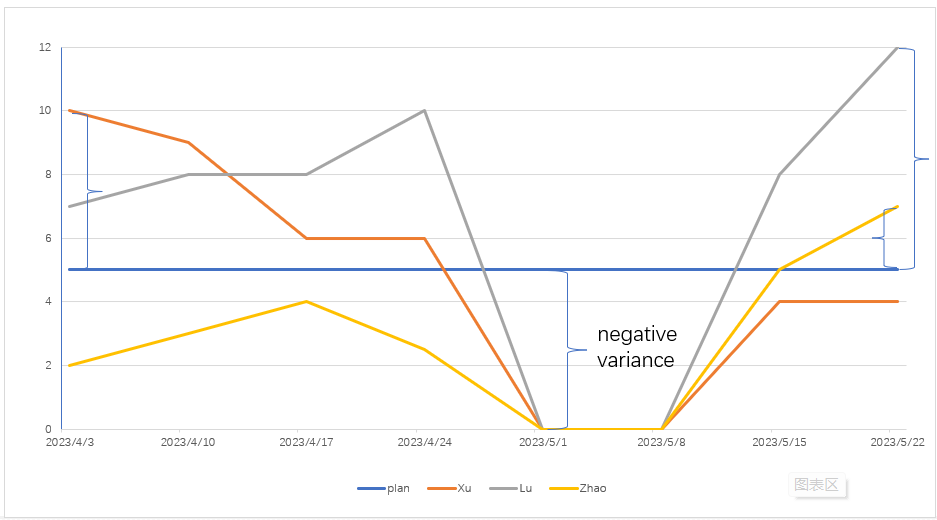


The data just displayed is all in years. Click to view the details and we can see data in months or even days. This is thanks to my team member Xu Wenjie's preprocessing operation on the data. You can see the calendar chart in the middle, and the pie chart next to it will update as the years change. On the far right is the line chart of the timeline for AQI.

# Chapter5 Project Execution Detail



The second part is the project execution details. The first diagram represents the execution process of the project, with the horizontal axis showing the completion time of each function and the vertical axis showing the project progress. The blue and red lines represent the planned and actual project progress, respectively. It can be seen that in the actual project progress, there was a period of time when the project did not progress, mainly because before and after the May Day holiday.From the graph, it can be seen that the maximum variance is as high as 20%.



The second chart represents the cost of time, the blue baseline represents the workload of five hours per week, and the other three curves represent the weekly workload of our team members. It can be seen that the workload of all three of us tends to be concentrated in the early and late stages of project implementation.As before, in terms of time cost, our variance during the holiday period is negative.

In summary, we have basically achieved our respective goals.

# Chapter6 Risk Plan

## 6.1 Introduction

The risk plan is useful for the whole project to establish the overall risk management objectives, coordinate the control of each member of the task, assess the risk level, and think of countermeasures in advance, which can make the project plan progress more smoothly.

## Project Risk Management Organization

Team 1 of project management

## Project Definition Risk Management Form

### Project Risk Level Definition

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk Level** | **Predicted Risks** | **Solution** | **Did we encounter** |
| Low-level Risk | Due to need to learn other subjects, some tasks cannot be completed as expected…… | Select other times to catch up the progress. | √ |
| During May Day, everyone was resting…… | √ |
| If someone leave of absence, the progress was delayed…… | √ |
| If someone gets sick, and can recover in the short term…… | × |
| Middle-level Risk | If someone gets sick, and need to takes a long time to recover…… | the only way to do so is to increase the workload of other members or to find new members. | × |
| High-level Risk | If the current plan is not feasible, or is not recognized by Party A (teacher) | Change the project plan. | × |
| If loss of the important data…… | Make backups in advance. | × |

### Project Risk Probability and Impact Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Qualitative Description** | **Probability** | **Influence** | **Progress Rate** | **Quality** | **Range** |
| Low Level | 0.7 | 0.2 | Progress delayed within three days | The quality is basically unaffected | 3 cases per week |
| Middle Level | 0.2 | 0.5 | Progress delayed by more than a week | The quality drops to the point where project plans need to be changed | 1 change in half of year |
| High Level | 0.1 | 0.8 | Progress delayed 15 days | The quality drops to the point where you have to restart the project | 1 major change in a year |

### Project Risk Status Definition

|  |  |
| --- | --- |
| **Risk Status** | **Status Description** |
| Tracking in progress | Under surveillance |
| Alleviating | The likelihood of the risk occurring may be reduced or the impact is being mitigated |
| Positive emergency handling | A risk beyond expectation has occurred and is being addressed urgently |
| Closed | Closed after successful risk response |
| Disappeared | The probability of risk occurrence may be reduced to zero |

## Project Risk Management Schedule

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | **Start** | **End** | **Responsible Person** | **Remarks** |
| Establish risk management team | 2023/3/23 | 2023/6/2 | Wenjie Xv、Qianlong Lu、Ruirui Zhao | None |
| Training on risk management | 2023/3/24 | 2023/3/26 | Wenjie Xv、Qianlong Lu、Ruirui Zhao | None |
| Define Risk Management Table | 2023/4/4 | 2023/6/1 | Ruirui Zhao | None |
| Identify and collect risks Round 1 | 2023/4/12 | 2023/4/13 | Ruirui Zhao | None |
| Identify and collect risks Round 2 | 2023/4/19 | 2023/4/20 | Ruirui Zhao | None |
| Identify and collect risks Round 3 | 2023/4/25 | 2023/4/26 | Wenjie Xv | None |
| Identify and collect risks Round 4 | 2023/5/4 | 2023/5/5 | Wenjie Xv | None |
| Identify and collect risks Round 5 | 2023/5/9 | 2023/5/10 | Qianlong Lu | None |
| Identify and collect risks Round 6 | 2023/5/22 | 2023/5/23 | Qianlong Lu | None |

# Chapter7 Test Plan & Test Cases

## 7.1 Test Plan

### 7.1.1 schedule

#### Test1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Content** | **Responsible Person** | **Start Date** | **End Date** | **Duration** |
| Test six-year interface to pollution averages by province in China | Xu Wenjie | 2023/4/20 | 2023/4/21 | 2 |

#### Test2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Content** | **Responsible Person** | **Start Date** | **End Date** | **Duration** |
| Test the interface to obtain the average value of pollutants for each province according to the year and the city pollution data according to the name of the province | Xu Wenjie | 2023/4/25 | 2023/4/26 | 2 |

#### Test3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Content** | **Responsible Person** | **Start Date** | **End Date** | **Duration** |
| Test interface to obtain provincial pollutant averages for the last six years and to obtain urban pollutant averages for the target year | Xu Wenjie | 2023/4/28 | 2023/4/29 | 2 |

#### Test4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Content** | **Responsible Person** | **Start Date** | **End Date** | **Duration** |
| Test to obtain pollutant data for the top ten provinces in the country for the last six years and interface to obtain the national average of pollutants for the last six years | Xu Wenjie | 2023/5/4 | 2023/5/5 | 2 |

#### Test5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Content** | **Responsible Person** | **Start Date** | **End Date** | **Duration** |
| Test the interface to obtain the pollutant data and average pollutant data of the top ten cities in a province for the last six years | Xu Wenjie | 2023/5/15 | 2023/5/16 | 2 |

### 7.1.2 Design Notes

#### 7.1.2.1 Test1

Interface：getAllProvincePollutions

Responsible Person：Xu Wenjie

##### Control

Testing with Spring Integration Junit.

##### Input

None

##### Output

Six-year average of each pollutant for all provinces.

#### 7.1.2.2 Test12

Interface：getInfoByYear、getCityInfoByProvince

Responsible Person：Xu Wenjie

##### Control

Testing with Spring Integration Junit.

##### Input

getInfoByYear：Year

getCityInfoByProvince：Year，Province

##### Output

getInfoByYear：Obtain the average value of pollutants for each province by year.

getCityInfoByProvince：Get pollution data for cities by year and province name.

#### 7.1.2.3 Test3

Interface：getSomeAvgCount、getSomeCityAvgCount

Responsible Person：Xu Wenjie

##### Control

Testing with Spring Integration Junit.

##### Input

getSomeAvgCount：None

getSomeCityAvgCount：City

##### Output

getSomeAvgCount：Average pollutants by province in the last six years.

getSomeCityAvgCount：Target year urban pollutant average.

#### 7.1.2.4 Test4

Interface：getTenProvinceAsc、getSixAverage

Responsible Person：Xu Wenjie

##### Control

Testing with Spring Integration Junit.

##### Input

getTenProvinceAsc：None

getSixAverage：None

##### Output

getTenProvinceAsc：Pollutant data for the top 10 AQI provinces for each of the six years.

getSixAverage：Six-year annual national average for each pollutant.

#### 7.1.2.5 Test5

Interface：getTenCityAsc、getSixAverageByProvince

Responsible Person：Xu Wenjie

##### Control

Testing with Spring Integration Junit.

##### Input

getTenCityAsc：None

getSixAverageByProvince：None

##### Output

getTenCityAsc：AQI pollutant data for the top 10 cities in a province for each of the six years.

getSixAverageByProvince：Average pollutant data for a province for each of the six years.

## 7.2 Test Case

### 7.2.1 Junit Introduction

The unit tests of this project are implemented through SpringBoot integration with Junit.

JUnit is a Java testing framework that makes it easy to write reliable and efficient tests. It can be used for applications made in most languages, but is particularly well suited for testing Java applications.JUnit can also be used to create automated tests.

The JUnit framework is one of the most popular Java testing frameworks. It provides several features that make writing tests easy, including support for multiple test cases, assertions, and reports. JUnit is also versatile, allowing tests to be written in a variety of languages.

It allows you to create and run tests efficiently and has become one of the most popular Java testing frameworks. xUnit framework inspired JUnit for Smalltalk and C++. Since JUnit is a member of the xUnit family of testing frameworks, it is designed to support different tests, including unit, functional and integration tests.

JUnit is primarily used for unit testing, but it can also be used for other tests such as functional and integration tests. Functional tests test the functionality of a system. They are different from unit tests because they test the entire system rather than individual units. Integration tests test the integration of two or more systems. They are different from unit tests because they test how the components of the system work together, rather than individually.

### 7.2.2 The way JUnit works

JUnit is a software testing framework that helps developers test their applications. It allows developers to write tests in Java and run them on the Java platform. JUnit also has a built-in reporter that prints out the results of the tests.

Automated testing with JUnit has two main goals. The first goal is to ensure that the software is doing what it is supposed to do. If a piece of code is supposed to do something and isn't doing it, you want to know as soon as possible so you can fix it. The second goal of automated testing with JUnit is to find bugs in your code; if you encounter problems with the way your code works, remember the golden rule: remove bugs before they are fixed!

There are several kinds of tests for JUnit programs. Unit tests test individual pieces of code in a class or method; integration tests, which test how all the pieces work together; and system tests, which test an entire system, such as a web server. When you have many tests, it is helpful to run them simultaneously; otherwise, they can take a long time to execute. You can use JUnit from the command line or in Eclipse.

JUnit provides several features that make it easy to create and run tests, including.

Assertions are used to verify the expected behavior of the system. JUnit provides a set of assertion methods that can be used to check the test results.

The test runner is used to execute tests and report the results, and JUnit provides a graphical test runner that can run tests and view the results.

Test suites are used to group related tests, and JUnit provides a way to create test suites that can be run together.

Reporting. When you run your tests, JUnit can help you analyze the results. It provides a built-in reporter that prints out information about the tests that were executed.

### 7.2.3 JUnit Benefits

There are many benefits to using JUnit, the most notable of which is its ability to help you develop reliable and testable code. Here are some other reasons why you'll definitely want to start using JUnit.

（1）JUnit can help you keep your code organized and easy to read.

（2）JUnit can help you detect and fix errors in your code.

（3）JUnit can help you improve the quality of your software.

（4）JUnit can help you work more efficiently and improve your testing process.

If you are a software developer and want to get started with JUnit, there are many resources to help you get started. You can find JUnit tutorials, documentation, forums, and more that can help you learn how to use JUnit effectively. with the right knowledge of the tools and support, you can start using JUnit to improve the quality of your code and make your development process more efficient.

### 7.2.4 Our JUnit Cases

We have a total of 19 test cases. The design of the test cases is divided into two parts, namely legal input and illegal input.

#### 7.2.4.1 Legal Input

By manually calculating the value that should be obtained by each method in advance and comparing it with the value calculated by the program, we determine whether the method obtains the correct result.

1. @Test
2. **public** **void** getTenProvinceAscTestProvince() {
3. String[] real2018TenProvinces = {"西藏", "青海", "黑龙江", "海南", "台湾", "内蒙古", "香港", "四川", "云南", "吉林"};
5. List<Map<String, Object>> tenProvinceAsc = pollutionService.getTenProvinceAsc();
6. // 获得省份名称
7. Map<String, Object> provinceMap = tenProvinceAsc.get(0);
8. **for** (Map.Entry<String, Object> entry : provinceMap.entrySet()) {
9. **if** (entry.getKey().equals("2018")) {
10. String[] test2018TenProvinces = (String[]) entry.getValue();
11. // 测试2018年省份排名是否正确
12. Assert.assertArrayEquals(test2018TenProvinces, real2018TenProvinces);
13. }
14. }
15. }

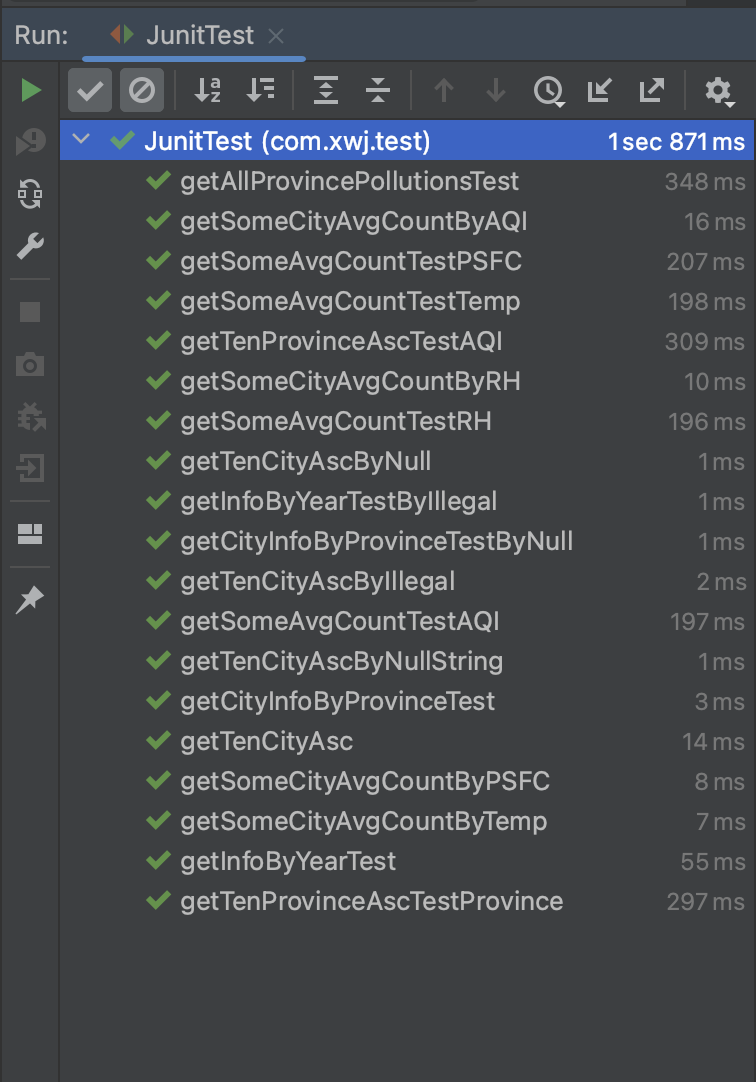
#### 7.2.4.2 Illegal Input

By feeding each method with illegal parameters, the methods are tested to see if they are robust enough to handle illegal parameters. The test passes if the returned result is the desired result. Illegal parameters include, null, empty strings and variables that are out of acceptable range.

1. @Test
2. **public** **void** getInfoByYearTestByIllegal() {
3. // 测试年份为null,空字符串,年份超出范围是否返回正常
4. List<Map<String, Object>> list = pollutionService.getInfoByYear(**null**);
5. Assert.assertNull(list);
6. list = pollutionService.getInfoByYear("");
7. Assert.assertNull(list);
8. list = pollutionService.getInfoByYear("2009");
9. Assert.assertNull(list);
10. }

#### 7.2.4.3 Test Results

All 19 of our test cases have been tested and passed successfully.



# Chapter8 Lesson Learnt

Plan the project progress in advance and set a good milestone, so that each person has a clear division of labor,predict the risks and countermeasures comprehensively, estimate the impact of each risk as it arises and how often it occurs, and try best to perform the task according to the plan.

Pay attention to communication, effective communication can make the project go smoothly. This not only gives us a clearer picture of how the project is progressing and generates better ideas, but also allows us to identify and respond to problems in a timely manner.

Do a good job of backing up and maintaining the project, because there is a possibility of losing important data in the risk plan and the impact is serious.

Understand the needs of users and market (job requirements) sufficiently, this allows us to avoid the risk of not being able to progress with the project and to get a better feedback.

# Chapter9 Conclusion

According to the project schedule, the visualization of pollutants in China and provinces between 2013 to 2018 was completed basically. Based on the results of visual analysis and assessment, the spatial and temporal distribution patterns and spatial and temporal evolution of air pollution over a six-year period were summarized.Most of the risks we encountered with have been predicted in advance ,and all the risks have been resolved.

The main manpower expenditure in the process of the project did not exceed the budget, and the planned functional points that need to completed were realized.

AmapNavi API call has limitations, go to the backup solution, use the database of geographical coordinate mapping done by others on GitHub, our data is large, and it took a long time to match the geographic coordinates of that database one by one, resulting in exceeding the time. In addition,there are still some minor problems in the system.

# Chapter10 Future Plan For Next Cycle

Perform multiple tests on the system and fix the current bugs

In the future, we plan to do secondary cache optimization by using local cache and Redis for second-level caching to avoid database downtime and system breakdown due to a large number of requests in high concurrency scenarios.

Do cluster regions with the change trend of pollutants in each quarter, and acquire the Spatial-temporal distribution model of air pollution through comparative analysis of space and time.