CS209A - Computer System Design and Applications A
2021F

Project Report

Backend: 张子研 11912324
Frontend: 韩梓辰 11910607

2021.12.22

1 Overview

In this project, we make three charts to show the worldwide and domestic data of COVID-19 pandemic cases.

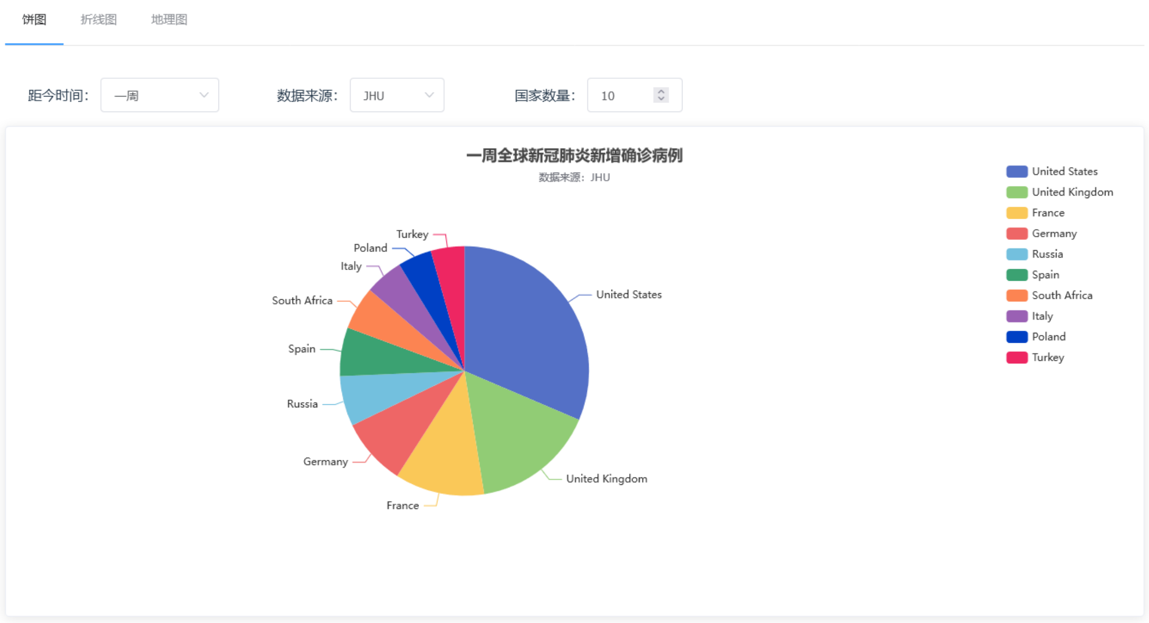
We have two data soures for world cases: JHU (Johns Hopkins University) and WHO (World Health Organization), and one data source for domestic data: Tencent News.

Backend part is for acquire data from open sources at regular intervals, load and process those data and return to frontend when required. For convenience, we use spring boot framework.

Frontend part is for get processed data from backend by api, and visualize those data. We use vue framework with element-ui to make layout and use echarts to achieve data visualization.

1.1 Pie Chart

A pie chart shows the amount of new cases confirmed in the past several days. Users can select data source, the past n days and the number of countries show in the chart, sorted by the new confirmed cases.



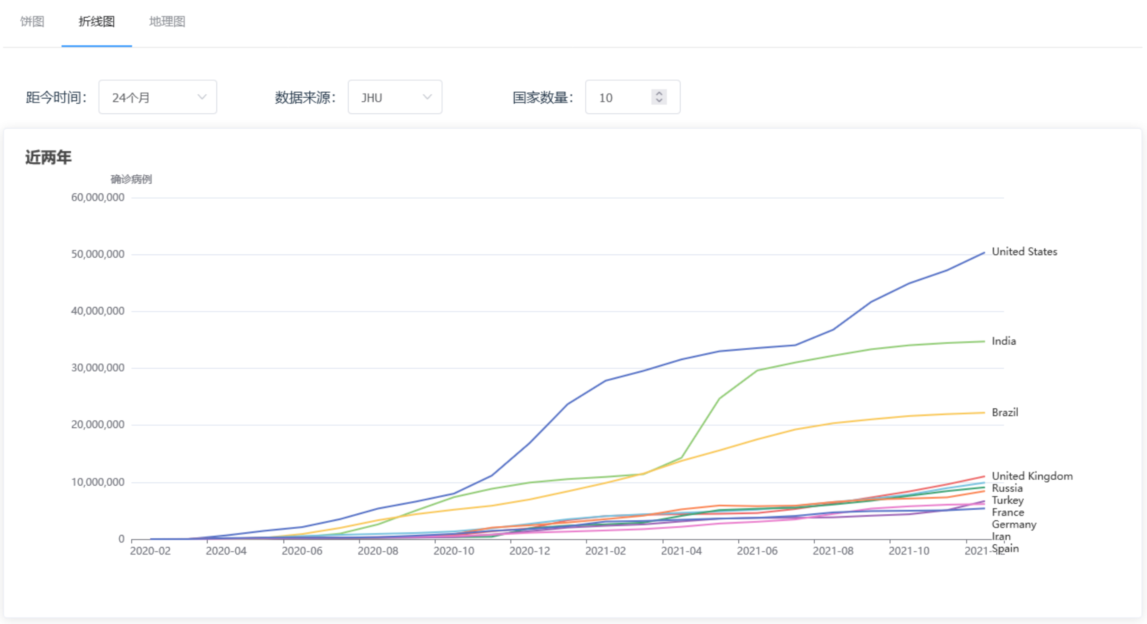
距今时间: 一周

数据来源: JHU

国家数量: 10

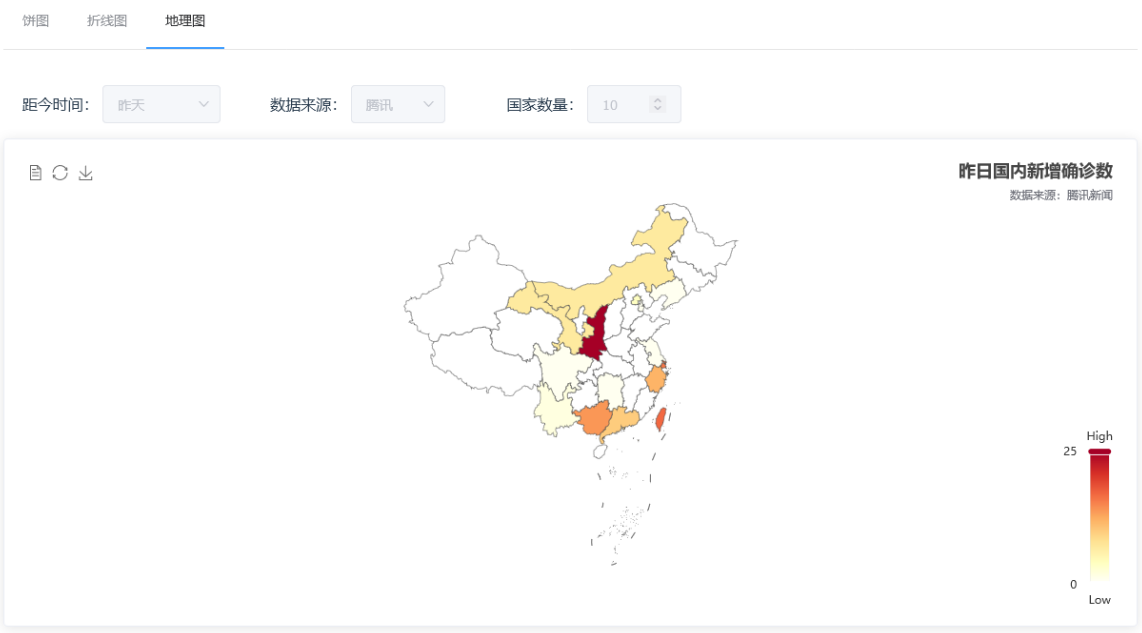
1.2 Line Race Chart

A line-race chart is a dynamic chart shows the trend of total cases from the beginning of this pandemic. Users can select data source and the number of countries show in the chart, sorted by the total case in this month.



1.3 China Map Chart

China map chart shows the amount of new confirmed cases yesterday, colored differently by number of cases.



1.4 Data Source

WHO data source: https://covid19.who.int/WHO-COVID-19-global-data.csv

JHU data source: https://raw.githubusercontent.com/owid/covid-19-data/master/public/data/jhu/full_data.csv

Tencent news data source: https://api.inews.qq.com/newsqa/v1/query/inner/publish/modules/list?modules=provinceCompare

2 Backend Design

2.1 Tree Structure

```
DataVisualizer
├─.idea
├─.mvn
│  └─wrapper
├─data
├─src
│  ├─main
│  │  ├─java
│  │  │  └─edu
│  │  │      └─sustech
│  │  │          └─datavisualizer
│  │  │              ├─controller
│  │  │              ├─entity
│  │  │              ├─task
│  │  │              └─utils
│  │  └─resources
│  │      └─static
│  │          ├─css
│  │          ├─fonts
│  │          └─js
│  └─test
│      └─java
│          └─edu
│              └─sustech
│                  └─datavisualizer
└─target
    ├─classes
    │  └─edu
    │      └─sustech
    │          └─datavisualizer
    │              ├─controller
    │              ├─entity
    │              ├─task
    │              └─utils
    ├─generated-sources
    │  └─annotations
    ├─generated-test-sources
    │  └─test-annotations
    └─test-classes
        └─edu
            └─sustech
                └─datavisualizer
```

In which, we store our COVID-19 pandamic data in the `./data` folder, all static files generated by frontend in `./src/main/resources/static`, all backend code in `./src/main/java/edu/sustech/datavisualizer`

2.2 Project Structure

In this part, I'll interpret functions and classes of backend.

```
datavisualizer
|  DataVisualizerApplication.java
|
├─controller
|      DataController.java
|      TestController.java
|
├─entity
|      CountryCase.java
|      StdData.java
|      TimeSeriesData.java
|
├─task
|      RenewSourceTask.java
|
└─utils
        DataLoader.java
        DataProcessor.java
        RequestData.java
```

2.2.1 DataVisualizerApplication.java

The Main class of Spring Application.

2.2.2 Controller Folder

2.2.2.1 DataController.java

1. `api/pieChart`: to acquire data for pie chart from backend, parameters include:

   source indicate the data source

   nDays indicate the new cases in the past n days

   n indicate only show the top n country with most cases

2. `api/lineRace`: to acquire data for line race chart from backend, parameters include:

   source indicate the data source

   n indicate only show the top n country with most cases

3. `api/mapChart`: to acquire data for china map chart from backend, no parameter.

2.2.2.2 TestController.java

A `/test/renew` api used only in test, for renew data from data source.

2.2.3 Entity Folder

2.2.3.1 CountryCase

An entity class contain 6 fields, used in raw data from the open source.

```java
@Data
public class CountryCase {
    private String country;
    private String date;
    private String totalCases;
    private String dailyCases;
    private String totalDeaths;
    private String dailyDeaths;
}
```

2.2.3.2 StdData

An entity class contain 2 fields, used in pie chart and map chart, indicate the case of a country or a province.

```java
@Data
public class StdData {
    private String name;
    private int value;
}
```

2.2.3.3 TimeSeriesData

An entity class contain 3 fields, used in line race chart, indicate the total case of a country in a month.

```java
@Data
public class TimeSeriesData {
    private String country;
    private String month;
    private int cases;
}
```

2.2.4 Task Folder

`RenewSourceTask.java` : to request the latest data from open sources per hour, and then load data from file into data structure in the program.

2.2.5 Utils Folder

2.2.5.1 DataLoader

Load and store data in those csv or json files into the following data structure.

```java
private static List<CountryCase> JHUData;
private static List<CountryCase> WHOData;
private static List<StdData> provinceData;
private static JSONObject chinaMap;
```

Specifically, chinaMap is a JSONObject indicate the map of china, which is used by frontend to draw the map of china.

2.2.5.2 DataProcessor

Invoked by apis in controller, process data from the List in DataLoader and return the result to the controller.

```java
public static List<StdData> newCasesInLastNDays(String source, int nDay, int n);
public static List<TimeSeriesData> lineRaceData(String source);
```

2.2.5.3 RequestData

Use for request data from online source and save in the `./data` folder

```java
public static boolean bySource(String source);
public static boolean getProvinceData();
```

`bySource`: to download worldwide pandemic data from WHO or JHU.

`getProvinceData`: to download domestic pandemic data by province from Tencent news.

3 Frontend Design

We used the Vue as our front-end template. And use ElementUI and Echart to build up our front-end.

3.1 Tree Structure

```
DataVisualizer
├─dist
├─public
|   ├─img.png
|   └─index.html
├─src
|   ├─components
|   |   ├─Charts.vue
|   ├─App.vue
|   └main.js
├.gitignore
├babel.config.js
├package.json
├package-lock.json
└README.md
```

3.2 Main Function in JS

At first we will initialize the page

```
mounted() {
    window.onresize = () => {
      this.mainCharts.resize();
    }
    this.changeCharts("pi");
  }
```

To make the page can resize, we overwritten the onresize function

For change N_Day and DataSource we will change the global variable and then rebuild the figure.

```
changeShown_DataSource(str){
        this.select_data_source = str;
        this.shown_data = str;
    },
    changeShown_N_Days(day){
      this.select_n_days = day;
      for (var i=0;i<this.date.length;i++){
        if (this.date[i].value === this.select_n_days){
          this.shown_days = this.date[i].label;
        }
      }
    },
    changeCountries(num){
      this.select_n_countries = parseInt(num)
      this.buildCharts(this.select_fig);
    },
    changeDataSource(str){
      this.changeShown_DataSource(str);
      this.buildCharts(this.select_fig);
    },
    changeNDays(day){
      this.changeShown_N_Days(day);
      this.buildCharts(this.select_fig);
    },
```

And rebuild.

```
buildCharts(figType){
      this.$forceUpdate();
      if(figType === "pi"){
        this.piCharts();
      }else if(figType === 'line'){
        this.lineCharts();
      }else {
        this.mapChart();
      }
    }
```

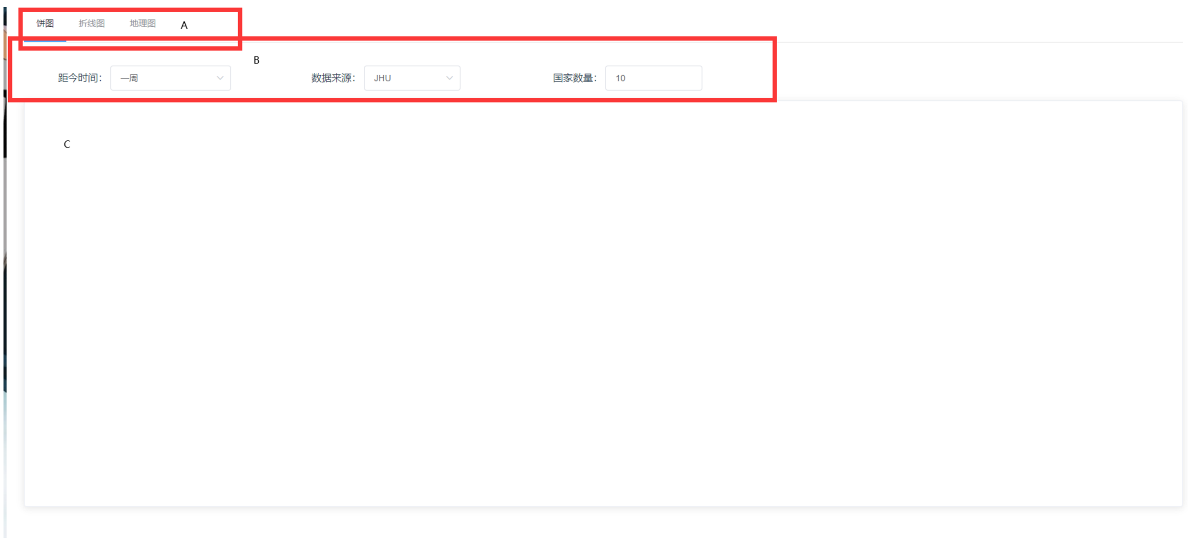
Every build we will get resource from back end. Here is an example shown how to get and build a fig.

```
piCharts(){
      this.$axios({
        method:'get',
        url:'api/pieChart',
        params:{
          source:this.select_data_source,
          nDays:this.select_n_days,
          n:this.select_n_countries,
        }
      }).then((response) =>{          //这里使用了ES6的语法
        response = response.data
        this.$echarts.init(document.getElementById('main')).dispose();
        this.mainCharts =
this.$echarts.init(document.getElementById('main'));
        var option = {
          title: {
            text: this.shown_days+"全球新冠肺炎新增确诊病例",
            subtext: '数据来源: '+this.select_data_source,
            left: 'center'
          },
          tooltip: {
            trigger: 'item',
            formatter: '{a} <br/>{b} : {c} ({d}%)'
          },

          legend: {
            type: 'scroll',
            orient: 'vertical',
            right: 10,
            top: 20,
            bottom: 20,
            data: response.legendData
          },toolbox: {
            show: true,
            //orient: 'vertical',
            left: 'left',
            top: 'top',
            feature: {
              dataView: {
                readOnly: true
              },
              restore: {},
              saveAsImage: {}
            }
          },
          series: [
            {
              name: '国家名',
              type: 'pie',
              radius: '55%',
              center: ['40%', '50%'],
              color:[
                '#5470c6',
                '#91cc75',
                '#fac858',
                '#ee6666',
                '#73c0de',
                '#3ba272',
                '#fc8452',
                '#9a60b4',
                '#0040c4',
                '#ee2662',
                '#91ac72'],
              data: response.seriesData,
              emphasis: {
                itemStyle: {
                  shadowBlur: 10,
                  shadowOffsetX: 0,
                  shadowColor: 'rgba(0, 0, 0, 0.5)'
                }
              }
            }
          ]
        };
        this.mainCharts.setOption(option);
        this.mainCharts.resize();
      }).catch((error) =>{
        console.log(error)       //请求失败返回的数据
      })
```

3.3 Basic page design

We split our page in three sections: the top menu bar A, the middle button bar B, and the main area icon display bar C.



All three parts are adaptive and work well with different screen sizes.

3.4 The main variables

Here are some main variables we use in our front-end.

```
        activeIndex:'1',
        shown_days:"一周",
        shown_data:"JHU",
        chart_state : 'bing',
        select_n_countries:10,
        select_data_source:"JHU",
        select_n_days:7,
        select_fig:"pi",
        disable_input:false,
        disable_n_day:false
```