武汉纺织大学

Web应用开发课程设计

**天气预报**

**学 院： 数学与计算机学院**

**班 级： 物联网11801**

**姓 名： 孙思萌**

**学 号： 1804240132**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年12月18日**

目 录

[1 需求分析 1](#_Toc59968863)

[1.1 数据来源 1](#_Toc59968864)

[1.2 数据呈现 1](#_Toc59968865)

[1.3各项功能 1](#_Toc59968866)

[1.3.1 切换城市 1](#_Toc59968867)

[1.3.2 显示当日天气 1](#_Toc59968868)

[1.3.3 显示未来七天天气 1](#_Toc59968869)

[1.3.4 绘制响应折线图 1](#_Toc59968870)

[1.3.5 根据气象情况动态改变背景图片 1](#_Toc59968871)

[1.3.6 根据更新时间动态改变背景图片 2](#_Toc59968872)

[1.3.7 背景图片淡入效果 2](#_Toc59968873)

[1.3.8 为背景图片添加遮罩层 2](#_Toc59968874)

[1.3.9 为输入框动态修改背景颜色 2](#_Toc59968875)

[1.3.10 动态改变天气图标 2](#_Toc59968876)

[1.3.11 各种图标的点击效果 2](#_Toc59968877)

[1.3.12 小贴士旋转过渡 2](#_Toc59968878)

[1.3.13 背景音乐 3](#_Toc59968879)

[2 系统实现 —— 后端 3](#_Toc59968880)

[2.1 项目结构 3](#_Toc59968881)

[2.2 配置文件 3](#_Toc59968882)

[2.3 控制器 4](#_Toc59968883)

[3 系统实现 —— 前端 6](#_Toc59968884)

[3.1 项目结构 6](#_Toc59968885)

[3.2 前端构建 7](#_Toc59968886)

[3.3 网络请求 8](#_Toc59968887)

[3.4 echart图表绘制 9](#_Toc59968888)

[4 系统测试 10](#_Toc59968889)

[5 系统总结 10](#_Toc59968890)

# 1 需求分析

设计一个天气预报页面，具体实现功能有：显示当日天气，显示七天天气，切换城市，绘制响应折线图等，具体细节如下：

## 1.1 数据来源

整个项目设计的数据来源于天气预报API：<http://www.tianqiapi.com/index/doc?version=v1>

这个API可以免费试用2000次，提供的数据也比较多，当然，如果掌握爬虫技术可以去中国气象局获取数据，中国气象局的数据是允许获取的。

## 1.2 数据呈现

当我们拿到数据之后，我们需要将数据呈现在页面上，有些页面可能是死数据，比如新闻信息，一旦加载出来就不会改变了，但是天气预报用到的数据是需要动态变化的，比如我们在页面上点击一下未来七天天气中的任何一天，我想要改变折线图，根据天气情况改变图片背景，那么在不刷新页面的情况下，就需要数据能够动态发生变化，这里用到的核心技术就是“响应式”，前端实现响应式我采用的是响应式框架“Vue”

## 1.3各项功能

### 1.3.1 切换城市

页面加载之初，页面上并没有切换城市的输入框，当用户点击“切换城市”之后，页面上出现输入框，用户在输入框中输入相应城市并敲击回车或者点击查找按钮之后，天气预报的城市自动切换为输入的城市。

### 1.3.2 显示当日天气

页面加载之初默认显示IP地址所在区域的天气情况，当用户通过“切换城市”功能切换城市之后，显示对应城市的当日天气。其中相应的小信息点，比如日出时间，日落时间，空气质量等。

### 1.3.3 显示未来七天天气

页面加载之初默认显示IP地址所在区域未来七天天气情况，当用户通过“切换城市”功能切换城市之后，显示对应城市未来七天的天气情况。

### 1.3.4 绘制响应折线图

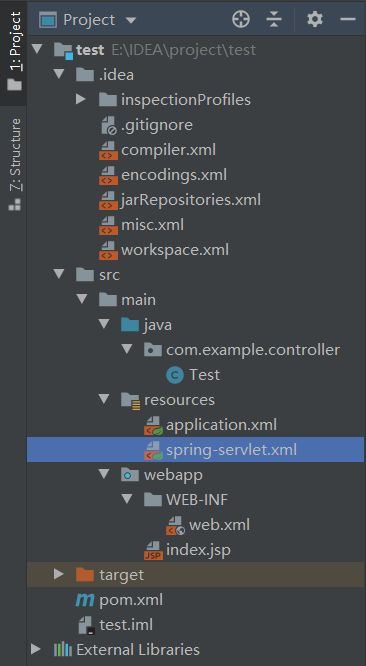
页面加载之初，通过echart绘制当前城市未来七天的气温趋势图，当用户点击未来七天预报的任意一天，则echart将绘制响应天的24小时气温变化图，当用户点击当日天气上的小信息点时，echart将绘制响应的信息变化图，如“未来七天的日出时间变化图”，“未来七天的日落时间变化图”，“未来七天的空气污染指数变化图”等等。

### 1.3.10 动态改变天气图标

根据不同的九种天气情况，找到九种匹配的天气图标，根据每天不同的天气情况动态选择对应的天气图标进行显示。

# 2 系统实现 —— 后端

## 2.1 项目结构



## 2.2 配置文件

在项目下新建Source Folder，名为resources，在创建application.xml配置文件

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans-3.0.xsd

http://www.springframework.org/schema/context

http://www.springframework.org/schema/context/spring-context-3.0.xsd ">

<!-- 下面是配置扫描包的位置,包名为com.ework.upms.server,也就是说,我们的试图解析器应该放在com.ework.upms.server包下. -->

<context:component-scan base-package="com.example.controller"/>

<bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">

<!-- 前缀,我们的视图文件应该放到/WEB-INF/views/目录下,这里我们需要在WEB-INF下面创建view文件夹 -->

<property name="prefix" value="/WEB-INF/views/" />

<!-- 设置后缀为.jsp -->

<property name="suffix" value=".jsp" />

</bean>

</beans>

## 2.3 控制器

发送请求，调用api接口，作为二传手将api的数据传送到前端

package com.example.controller;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.\*;

import java.io.\*;

import java.net.\*;

import java.util.Map;

@Controller

@CrossOrigin

public class Test {

@RequestMapping(value = "weather.api",method = RequestMethod.GET)

@ResponseBody

public String test(@RequestParam (value = "city",required = false, defaultValue = "")String city) throws IOException {

//外接口路径

String urlStr="http://v0.yiketianqi.com/api";

//get请求参数拼接

String params = "?appid=22266748&appsecret=pTIzTk6N&version=v9&vue=1";

String cityDecoded = "";

if(city != "") {

cityDecoded = "&city="+URLEncoder.encode(city,"UTF-8");

}

String urlStrFinal = urlStr + params + cityDecoded;

StringBuffer result = new StringBuffer();

URL url = new URL(urlStrFinal);

URLConnection urlConnection = url.openConnection();

InputStream inputStream = urlConnection.getInputStream();

BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(inputStream));

String line = "";

while((line = bufferedReader.readLine()) !=null){

result.append(line);

}

bufferedReader.close();

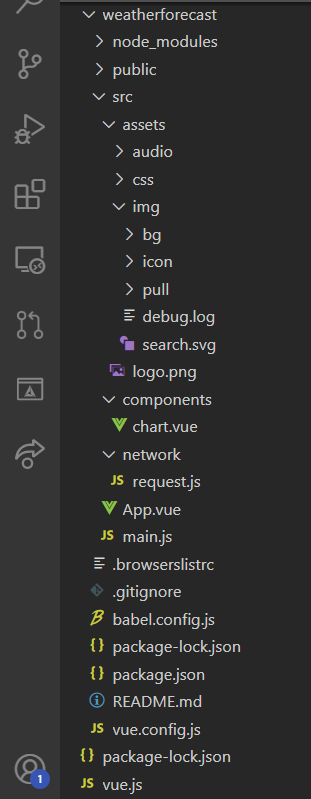
return result.toString();

}

}

# 3 系统实现 —— 前端

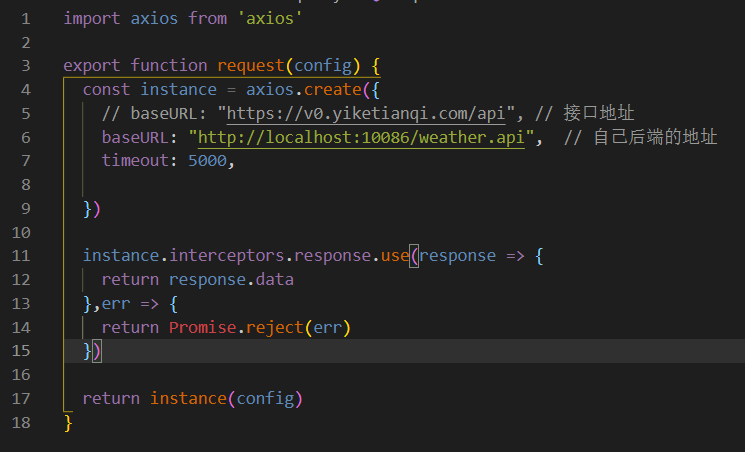
## 3.1 项目结构



## 3.2 前端构建



## 3.3 网络请求

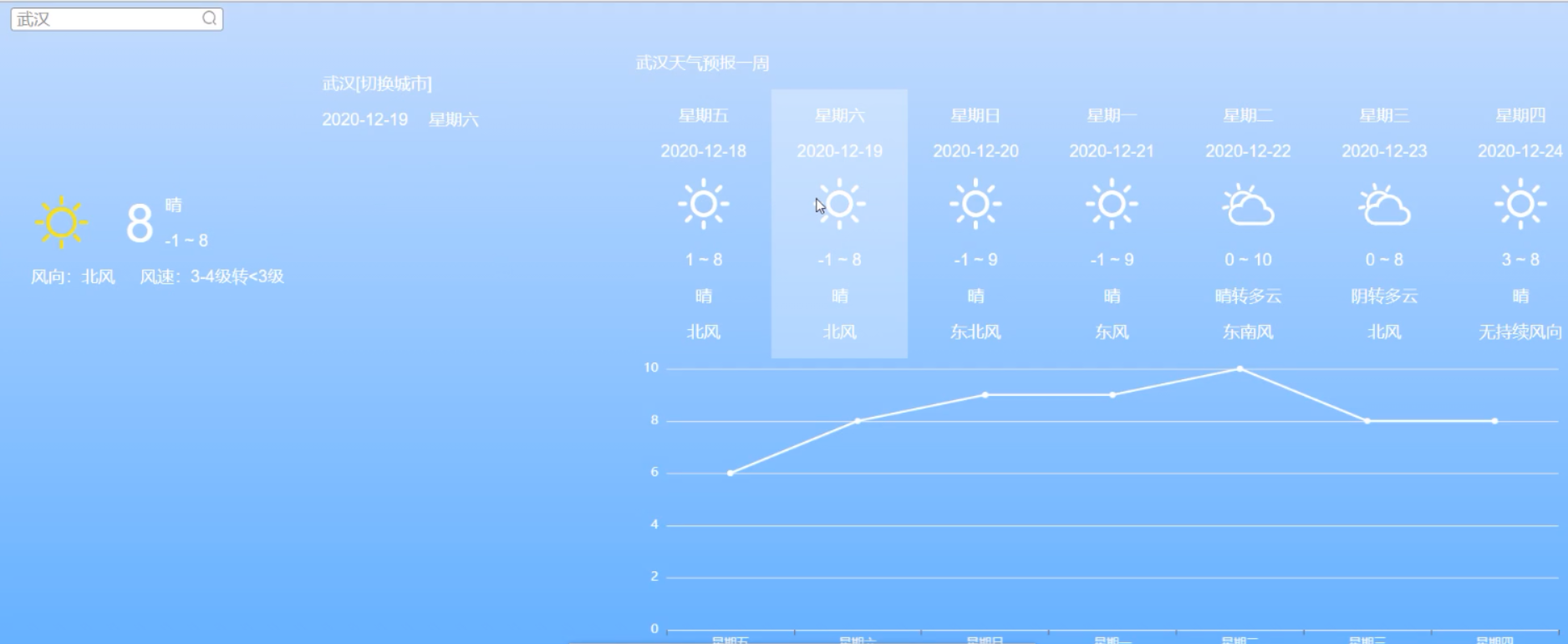


## 3.4 echart图表绘制



# 4 系统测试

测试结果



# 

# 5 系统总结

主要问题：

前端方面想要给用户以更好的使用体验，但不是很容易实现。

解决过程：

通过查阅相关的资源资料，请教了掌握得比较好的同学，受益匪浅。

系统存在的不足：

背景动画显示转换速度较慢、背景音乐无法实现自动播放等。

课程的收获：

终于感觉到可以自己写一些东西了，很有成就感，从聂老师那里学到的远不止课堂里的那么多。

发展方向：

因为自己的基础不是很好，目前已经在准备考研了，数学英语已经纳入了每天的学习计划之中。

自评成绩：

做这次课设说实话对我的挑战还是蛮大的，但是在我的不懈努力之下，还是能够完成老师对本课题的基本要求，如果要给自己打个分的话，我给自己85分吧，每次的课后作业和最后的大作业我都有很花心思地去完成，当然这些在聂老师的辛苦付出面前不值一提。

最后，我还是要感谢聂刚老师，并且为自己能够遇到聂老师这样认真负责的老师而心存感激！