

# 《地理信息工程软件开发》

## 总结报告

学 院: 遥感信息工程学院

班 级: 1705 C7

学 号: 2017302590132

姓 名: 姜屿涵

指导教师: 孟庆祥

2019 年 11 月 9 日

# 目录

一. 概述.....	4
1.1 实习目标 .....	4
1.2 实习内容 .....	4
二. 开发环境.....	4
2.1 开发工具和开发语言.....	4
2.1.1 开发工具.....	4
2.1.2 辅助开发工具.....	6
2.1.3 开发语言.....	6
2.2 技术栈.....	7
2.2.1 前端.....	7
2.2.2 后端.....	8
三. 功能设计.....	9
3.1 系统架构图.....	9
3.2 功能模块结构图.....	10
3.3 功能描述.....	10
3.3.1 数据管理模块.....	10
3.3.2 数据可视化模块.....	11
3.3.3 数据分析模块.....	12
3.3.4 系统维护模块.....	22
四. 开发流程.....	23
4.1 开发设计流程图 .....	23
4.2 软件安装与环境配置.....	24
4.2.1 前后端编译器的安装 .....	24
4.2.2 前端的开发工具 .....	24
4.2.3 后端的开发工具 .....	25
4.2.4 数据库的安装配置 .....	26
4.2.5 三维可视化软件的安装 .....	26

4.3 创建项目.....	27
4.3.1 前端项目 .....	27
4.3.2 后端项目 .....	27
4.3.3 创建数据库 .....	28
4.4 项目编写.....	29
4.4.1 数据库编辑 .....	29
4.4.2 后台开发 .....	30
4.4.3 前端开发 .....	36
五. 软件介绍.....	40
5.1 功能点.....	40
5.1.1 二维地图 .....	40
5.1.2 二维专题地图 .....	41
5.1.3 三维地图 .....	41
5.1.3 路线地图 .....	42
5.1.4 三维地球 .....	42
5.1.5 词云 .....	43
5.1.6 折线图 .....	43
5.1.7 图片和文字展示 .....	44
5.1.8 三维模型 .....	44
5.2 基本操作.....	45
5.3 程序运行效果.....	46
5.3.1 前端运行效果 .....	46
5.3.2 后台运行效果 .....	48
5.4 关键代码.....	48
5.4.1 前端关键代码 .....	48
5.4.2 后端关键代码 .....	50
六. 总结.....	51
6.1 关于系统本身.....	52

6.2 关于小组合作.....	52
七. 软件特色及改进建议.....	53
7.1 软件特色 .....	53
7.2 问题及建议 .....	54

# 一. 概述

## 1.1 实习目标

通过学习 Arc Engine 二次开发或使用其他地理信息处理工具，进行 GIS 软件开发并得出成果，提高学生的动手操作能力和编程能力。

通过小组合作，增强学生的合作意识，提高沟通交流与合作能力。

通过 PPT 展示小组成果，提高学生表达能力。

## 1.2 实习内容

以小组为单位，使用 Arc Engine 进行二次开发，或自主进行 GIS 软件开发，通过调用不同平台和数据库、语言学习、代码编写和程序运行实现对地理信息的综合处理和应用，开发一个完整的 GIS 软件。本次实习中，本小组使用其它地理信息处理工具进行开发。

# 二. 开发环境

## 2.1 开发工具和开发语言

### 2.1.1 开发工具

#### 1. IntelliJ IDEA:



IDEA 全称 IntelliJ IDEA，是 java 编程语言开发的集成环境。IntelliJ 在业界被公认为最好的 java 开发工具，尤其在智能代码助手、代码自动提示、重构、J2EE 支持、各类版本工具(git、svn 等)、JUnit、CVS 整合、代码分析、创新的 GUI

设计等方面的功能可以说是超常的。它的旗舰版本还支持 HTML, CSS, PHP, MySQL, Python 等。

在本项目中主要用于前端和后台的搭建。

## 2. 3ds Max:



3D Studio Max, 常简称为 3d Max 或 3ds MAX, 是 Discreet 公司开发的（后被 Autodesk 公司合并）基于 PC 系统的三维动画渲染和制作软件。

在本项目中主要用于地形的三维建模。

## 3. MySQL:



关系型数据库管理系统。MySQL 是最流行的关系型数据库管理系统之一，在 WEB 应用方面，MySQL 是最好的 RDBMS (Relational Database Management System, 关系数据库管理系统) 应用软件之一。

## 4. Navicat:



一套快速、可靠并价格相当便宜的数据库管理工具，拥有图形用户界面，是专为简化数据库的管理及降低系统管理成本而设的。

## 5. JDK:



JDK 是 Java 语言的软件开发工具包，主要用于移动设备、嵌入式设备上的 java 应用程序。JDK 是整个 java 开发的核心，它包含了 JAVA 的运行环境 (JVM+Java 系统类库) 和 JAVA 工具

## 6. Maven:



Maven 是一个项目管理工具，可以对 Java 项目进行构建、依赖管理。

## 2.1.2 辅助开发工具

### 1. Postman:



Postman 是用于 API 开发的协作平台。Postman 的功能简化了构建 API 的每个步骤并简化了协作，因此您可以更快地创建更好的 API。

在本项目中主要用于，调试后端代码，查看 http 和 https 的 GET/POST/PUT 请求的直接结果。

### 2. Google Chrome:



Google Chrome 是由 Google 开发的一款设计简单、高效的 Web 浏览工具。

在本项目中主要是使用其自带的开发者选项，调试模块，进行前端代码的调试。

## 2.1.3 开发语言

### 1. HTML+CSS:



超文本标记语言（HTML）是一种用于创建网页的标准标记语言。

层叠样式表是一种用来表现 HTML（标准通用标记语言的一个应用）或 XML（标准通用标记语言的一个子集）等文件样式的计算机语言。CSS 不仅可以静态地修饰网页，还可以配合各种脚本语言动态地对网页各元素进行格式化。

在本项目中，主要用于编写前端 Web 样式。

### 2. JavaScript:



JavaScript 是一种属于网络的脚本语言，已经被广泛用于 Web 应用开发，常用来为网页添加各式各样的动态功能，为用户提供更流畅美观的浏览效果。

在本项目中，主要用于 Echarts 图表的绘制，编写动态效果以及实现 HTTP 请求，获取后台提供的数据。

### 3. Java:



Java 是一门面向对象编程语言，不仅吸收了 C++ 语言的各种优点，还摒弃了 C++ 里难以理解的多继承、指针等概念，因此 Java 语言具有功能强大和简单易用两个特征。Java 可以编写桌面应用程序、Web 应用程序、分布式系统和嵌入式系统应用程序等。

在本项目中，主要用于后台服务器的搭建。

### 4. SQL



结构化查询语言 (Structured Query Language) 简称 SQL，是一种特殊目的的编程语言，是一种数据库查询和程序设计语言，用于存取数据以及查询、更新和管理关系数据库系统。

## 2.2 技术栈

### 2.2.1 前端

#### 1. Bootstrap:



Bootstrap 是美国 Twitter 公司的设计师 Mark Otto 和 Jacob Thornton 合作基于 HTML、CSS、JavaScript 开发的简洁、直观、强悍的前端开发框架，使得 Web 开发更加快捷。Bootstrap 提供了优雅的 HTML 和 CSS 规范，它即是由动态 CSS 语言 Less 写成。

#### 2. Echarts:



ECharts，一个使用 JavaScript 实现的开源可视化库，可以流畅的运行在 PC 和移动设备上，兼容当前绝大部分浏览器（IE8/9/10

/11，Chrome, Firefox, Safari 等），底层依赖矢量图形库 ZRender，提供直观，交互丰富，可高度个性化定制的数据可视化图表。

### 3. jQuery:



具有独特的链式语法和短小清晰的多功能接口；具有高效灵活的 css 选择器，并且可对 CSS 选择器进行扩展；拥有便捷的插件扩展机制。

### 4. AJAX:



AJAX = 异步 JavaScript 和 XML。AJAX 是一种用于创建快速动态网页的技术。通过在后台与服务器进行少量数据交换，AJAX 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

本次实习使用 Bootstrap 模板，搭建前端的基本样式。利用 Echarts，实现二维以及三维图表和地图的绘制，利用 JQuery, AJAX 实现与服务器交换数据。

## 2.2.2 后端

### 1. Spring Boot:



Spring Boot 是由 Pivotal 团队提供的全新框架，其设计目的是用来简化新 Spring 应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。另外 Spring Boot 通过集成大量的框架使得依赖包的版本冲突，以及引用的不稳定性等问题得到了很好的解决。

### 2. JPA:

JPA 是 Java Persistence API 的简称，中文名 Java 持久层 API，是 JDK 5.0 注解或 XML 描述对象—关系表的映射关系，并将运行期的实体对象持久化到数据库中。

### 三. 功能设计

#### 3.1 系统架构图

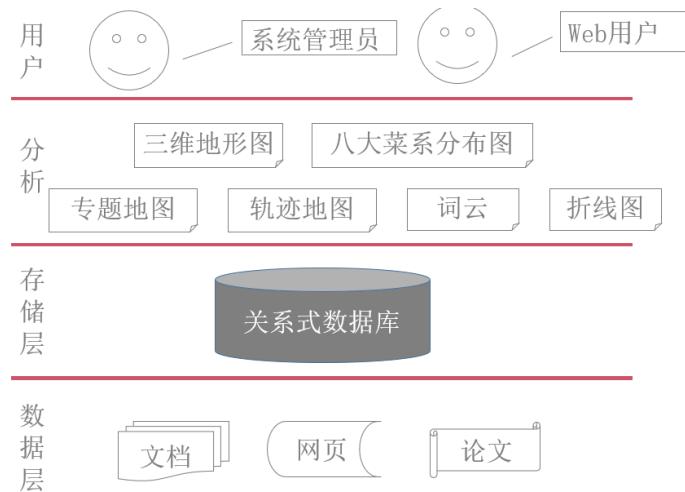


图 3.1 系统架构图

## 3.2 功能模块结构图

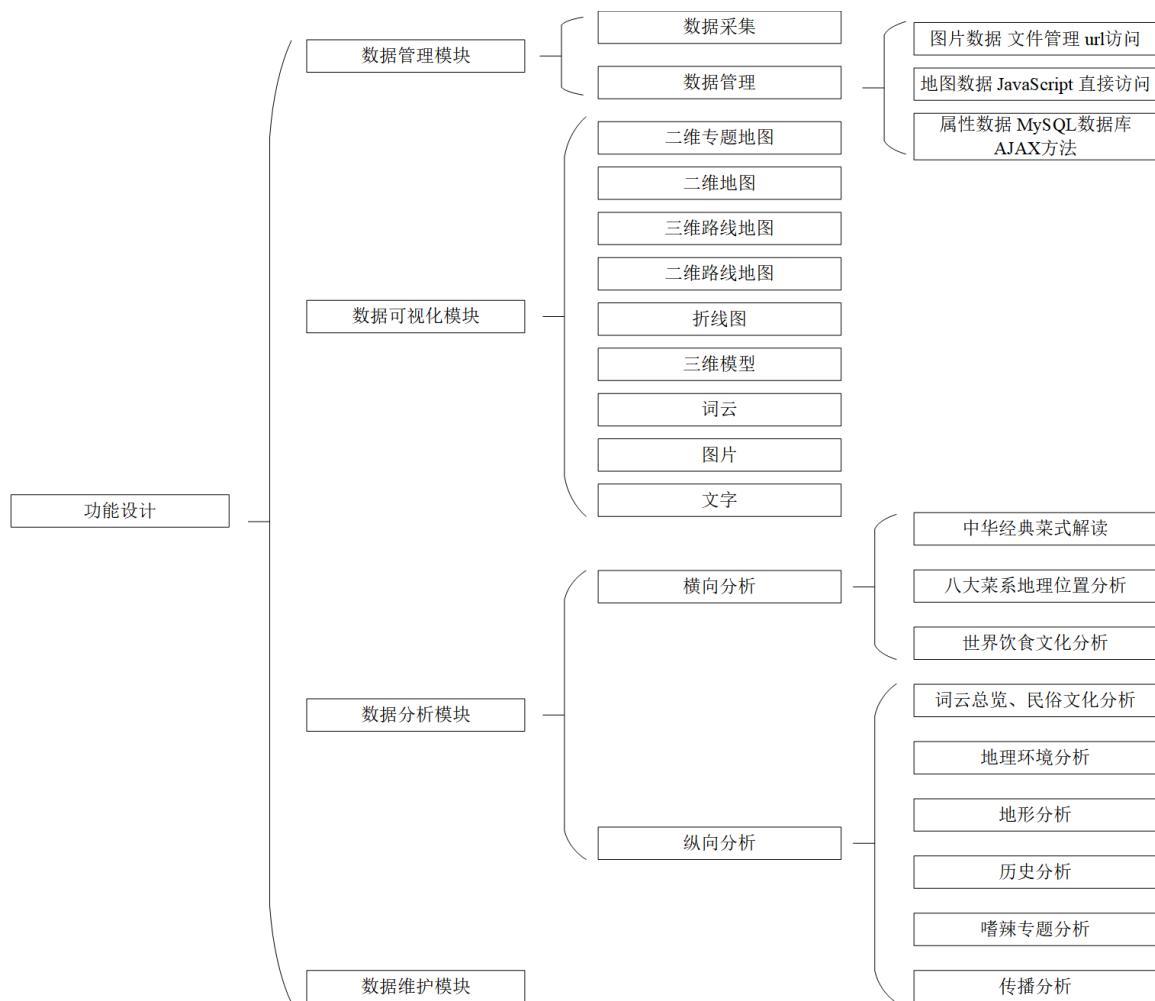
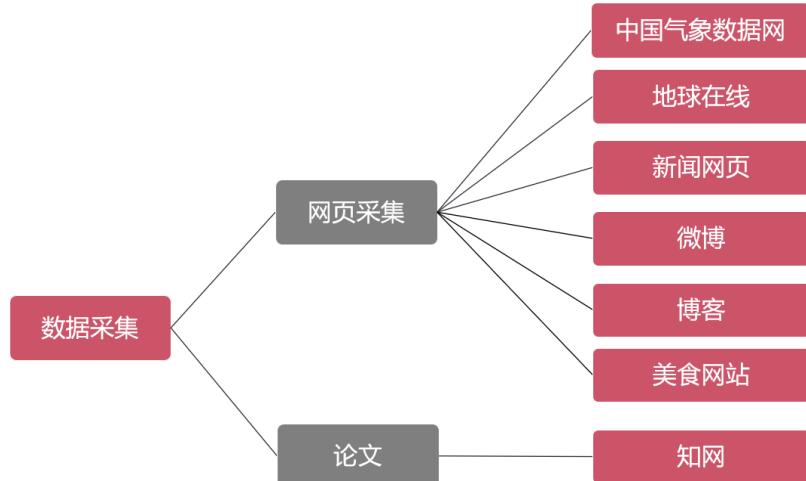


图 3.2 功能模块结构图

## 3.3 功能描述

### 3.3.1 数据管理模块

#### 1. 数据采集



## 2. 数据管理

数据是 GIS 的血液，对于数据的管理则是一个 GIS 平台的底层任务，所以需要一个模块进行数据管理。平台数据分为三部分，地图数据，图片数据，其它属性数据。数据管理方式是关系-对象数据库的管理方式，以省份为对象，对象的空间信息封装存储在 JavaScript 中，对象的属性数据存储在 MySQL 数据库中，关于描述对象的图片，将图片的 url（本地图片和网络图片）存入 MySQL 数据库中。

数据类型	数据管理方法	数据访问方式
图片	文件管理（本地图片）	URL
地图	JavaScript	直接访问
属性数据	MySQL 数据库	AJAX 方法

表 3.1 数据管理

### 3.3.2 数据可视化模块

这一模块对基础数据和分析结果进行可视化显示。对于地图，我们采用 2D 和 3D 两种方式显示。属性数据的显示包括，关键词云显示，图片和文字描述，在网

页中根据用户的需求进行显示。分析结果的可视化，包括专题地图，路线地图，变化趋势曲线。

数据类型	可视化方式
八大菜系分布	二维地图、图片、文字
历史传播数据	三维路线地图、文字
嗜辣程度	二维专题地图、数字
辣椒传播数据	二维路线地图、三维路线地图
地理环境数据	折线图
地形	三维模型
关键词	词云、图片
Food Map	图片
经典菜式	图片

表 3.2 数据可视化

### 3.3.3 数据分析模块

项目从横向和纵向两个方向，结合地理、民俗等多个方面，对美食文化的产生传播进行了分析，包括历史分析、地理分析、传播路线分析、由点及面分析、辣度专题分析。横向主要是八大菜系的形成与全球饮食文化形成与地理环境的关系，纵向主要是针对一个菜系，深入的分析其饮食文化的形成，包括历史、民俗等方面。以四川为例，体现全球范围内美食与地理密不可分的关系。

#### 1. 横向分析

- 1) 中华经典菜式解读：以书页的形式展现，寓意中华饮食文化丰富多样，博大精深，是一本耐人寻味的书，值得我们细细品读学习。同时以文字和图片结合的方式，向用户介绍菜品形成的历史故事。



图 3.4 满汉全席

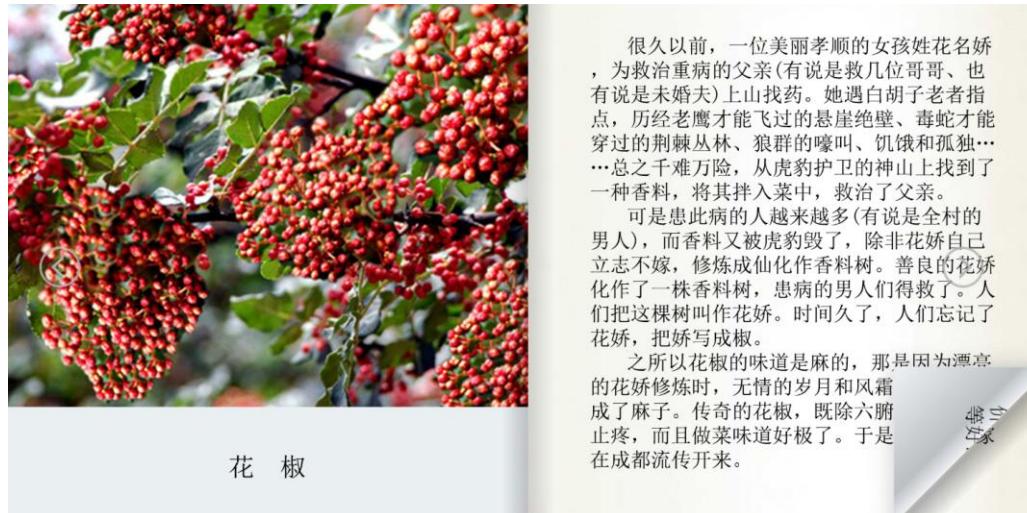


图 3.5 花椒

- 2) 八大菜系地理位置分析：采用地图的方式，在空间上展示八大菜系，将八大菜系的形成与其所处的地理位置相结合进行分析，同时结合图片展示和文字叙述的方式介绍八大菜系。

满汉全席这种消费方式源于清代，这中间还有一段传奇故事，我们知道乾隆皇帝一生风流倜傥，吃喝玩乐概括他的一生，七下江南，看遍江南美景，更吃了江浙等地美食，他一生虽然爱玩，但他是一个孝子，所以对父母十分关心。

正好太后要过60大寿，当时满汉人并不团结，很多地方出现反清志士，乾隆为了让满汉一家亲，于是让满汉有名的厨师做精品菜，这样意义十分深刻，可以让满汉人团结在一起，更让大家尝到满汉两族的美食。用这种特殊方式来表达满汉一家亲，这真难为乾隆皇帝。

满汉全席在普通人眼里可望而不可及，那满全席到底有多少种菜？有哪些地方的美食？满汉全席主要以东北、山东、北京、江浙菜为主，这些被称为满汉全席中的珍品，满汉全席共108道菜，南菜54道：30道江浙菜，12道闽菜，12道广东菜。北菜54道：12道满族菜，12道北京菜，30道山东菜。

很久以前，一位美丽孝顺的女孩姓花名娇，为救治重病的父亲（有说是救几位哥哥、也有人说他是未婚夫）上山找药。她遇白胡子老者指点，历经老鹰才能飞过的悬崖绝壁、毒蛇才能穿过的荆棘丛林、狼群的嚎叫、饥饿和孤独……总之千难万险，从虎豹护卫的神山上找到了一种香料，将其拌入菜中，救治了父亲。

可是患此病的人越来越多（有说是全村的男人），而香料又被虎豹毁了，除非花娇自己立志不嫁，修炼成仙化作香料树。善良的花娇化作了一株香料树，患病的男人们得救了。人们把这棵树叫作花娇。时间久了，人们忘记了花娇，把娇写成椒。

之所以花椒的味道是麻的，那是因为漂亮的花娇修炼时，无情的岁月和风霜成了麻子。传奇的花椒，既除六腑止疼，而且做菜味道好极了。于是在成都流传开来。



图 3.6 八大菜系分布图



图 3.7 图片和文字介绍

- 3) 世界饮食文化分析：使用由食物组成的世界地图，展示世界各国各国家不同的饮食文化。如美国爱吃玉米小麦等食物，其地图则由玉米

小麦组成，而澳大利亚，由于四面环海，拥有丰富的海产品，其地图由海鲜组成，而印度的香料也十分著名，印度地图则有香料组成。



图 3.8Food Map



图 3.9 美国

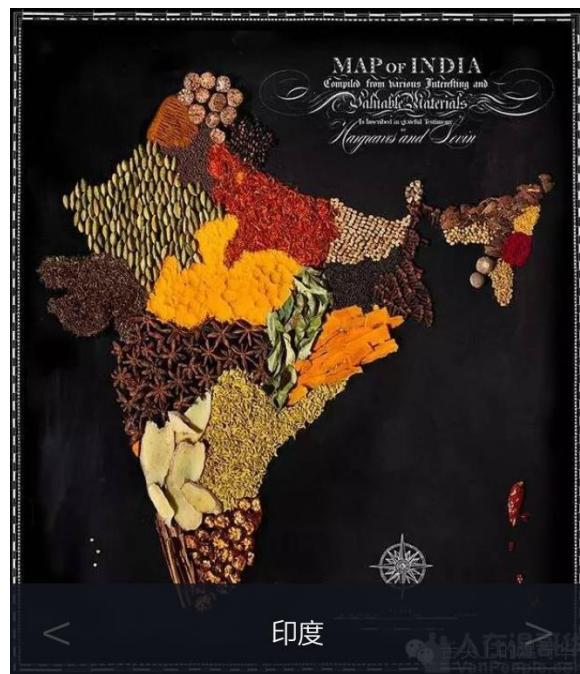


图 3.10 印度



图 3.11 澳大利亚

## 2. 纵向分析（川菜）

- 1) 词云总览：从全局的角度，结合民俗、文化形成的因素，通过词云分析目前川菜的情况，结合相关图片对菜品的介绍，增加用户对菜品的了解认识。如全国闻名的郫县豆瓣，全国连锁火锅店，小龙坎、海底捞等，能够让用户在关键词中看到自己熟悉的名字，可以直接感受到

川菜在全国范围内的传播；如火锅蛋糕、红油冰淇淋等，可以看到川菜的如今的发展，已经超出了典型的菜式，而是与多种其余食物相结合，产生了饮食文化的交融；如普通人家做饭的调料、伤心凉粉等，可以看到四川人吃辣的确是一种习惯，在普通家庭中辣椒是必不可少的调料，在菜品的命名中也可以看到辣的影子。



图 3.12 词云总览



图 3.13 火锅蛋糕和红油冰淇淋



图 3.14 调料和海底捞

## 2) 地理因素分析:

- 从地理环境的角度，分析四川吃辣文化的形成于地理环境的关系。主要是从温度和湿度两个角度进行分析。发现四川由于湿热的气候，人们需要通过吃辣椒排湿，因此形成了四川独特的吃辣文化。

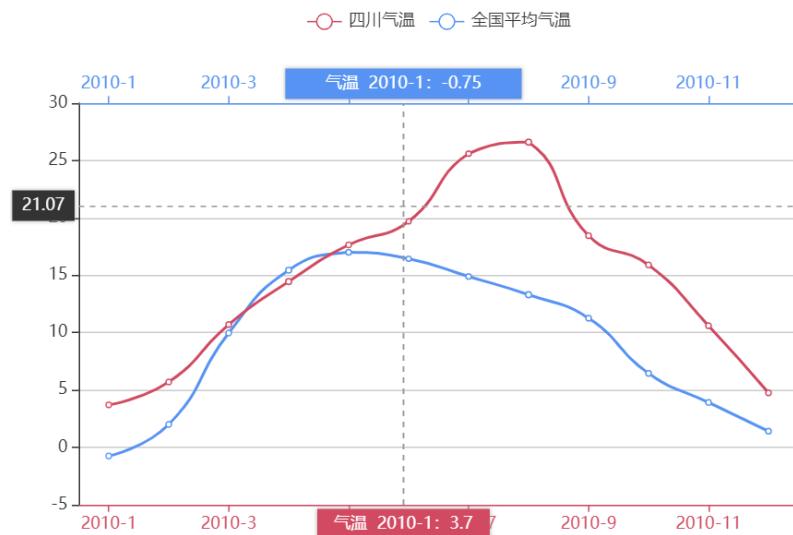


图 3.15 温度曲线对比图



图 3.16 湿度曲线对比图

- 地形分析：建立四川的三维地形图，从地形的角度，分析四川湿热气候形成的原因。通过分析发现，由于四川地处盆地，西邻青

藏高原，暖气流进入盆地后，就会被高原阻挡，因此当地湿气较重，为除湿器，当地人便喜爱吃辣。

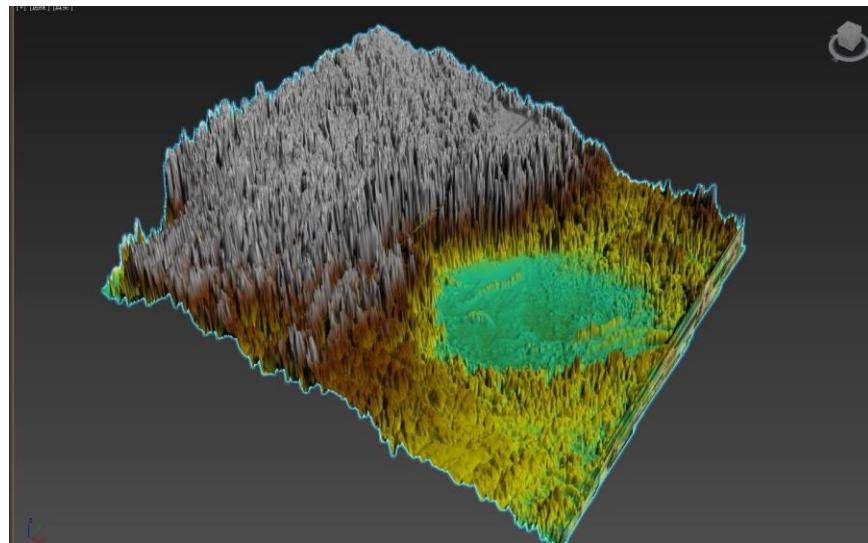


图 3.17 四川三维地形

- 3) 历史分析：从辣椒传播历史的角度，结合世界地图，分析辣椒如何传入我国，最后形成吃辣的文化。辣椒的传播主要是以典型的辣带为主。而辣椒传入我国，则是由著名的海上丝绸之路以及陆上丝绸之路传入。丝绸之路为我们引入了众多食材调料，极大的丰富了我国的饮食文化。

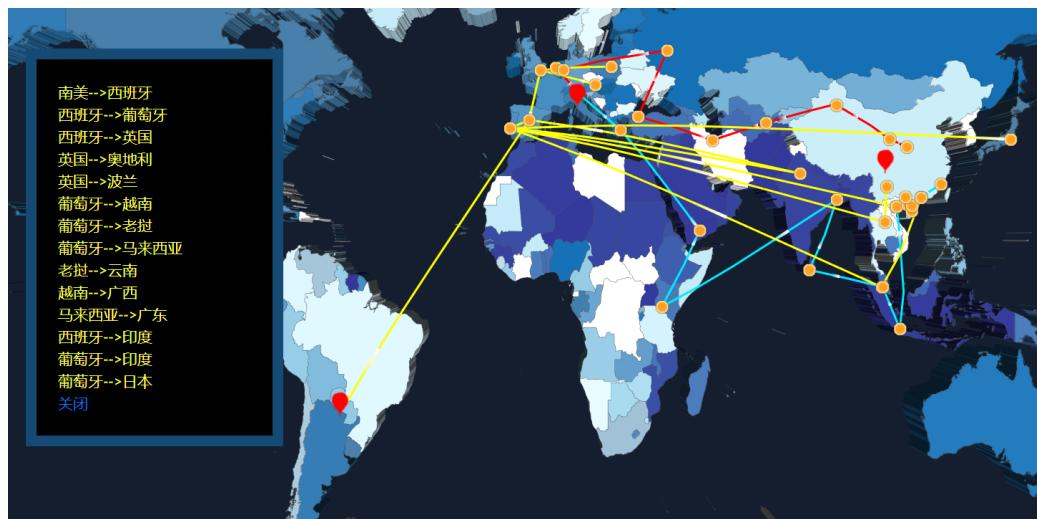


图 3.18 辣带

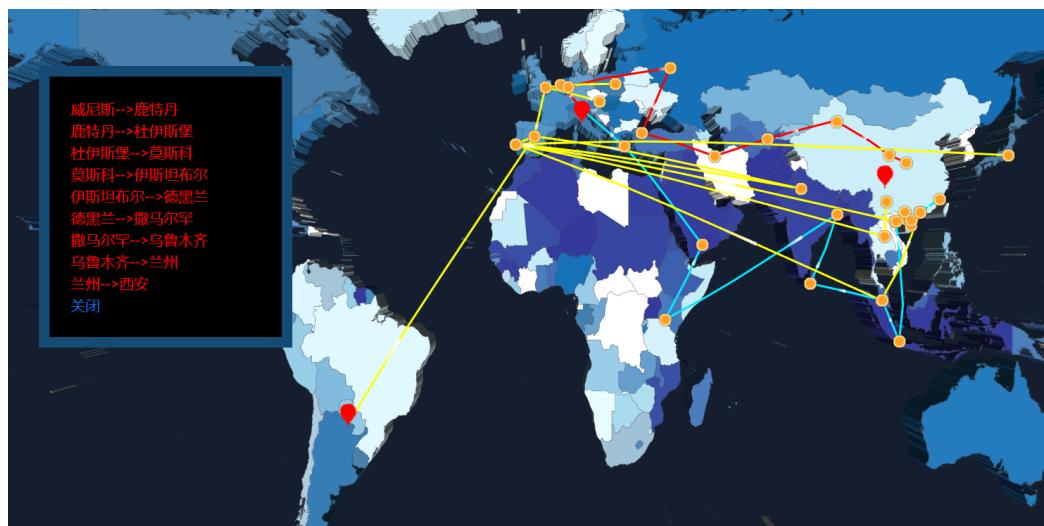


图 3.19 陆上丝绸之路



图 3.20 海上丝绸之路

- 4) 专题分析：以辣椒为专题，制作嗜辣专题地图，结合地理位置，分析全国范围的吃辣情况，发现形成了以长江流域为主的吃辣带，而福建沿海一带则不嗜辣。



图 3.21 嗜辣专题地图

5) 传播分析：通过，地理因素分析、历史分析、专题分析，了解四川食辣文化的形成因素后，对当前吃辣文化的传播进行分析。形成以四川为中心向全国传播，最后向全球传播的趋势。



图 3.22 全国传播

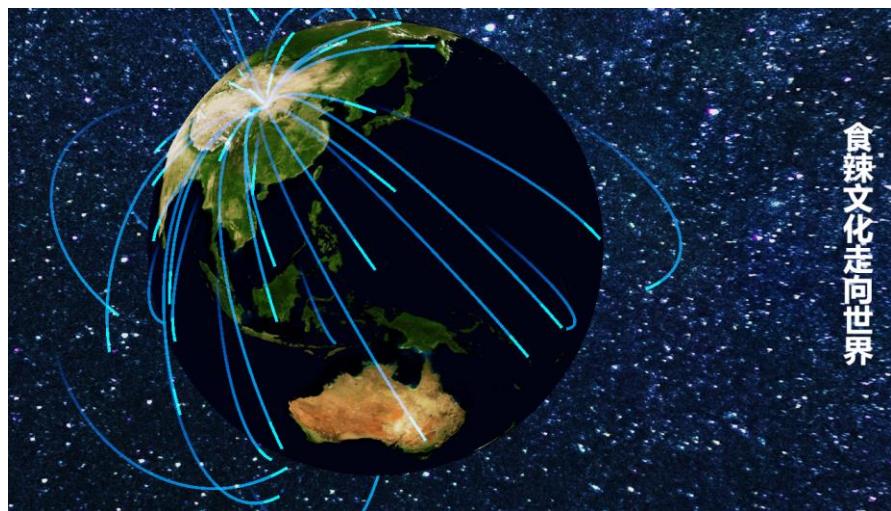


图 3.23 全球传播

### 3.3.4 系统维护模块

该模块主要是对系统稳定性，信息安全的维护。采用前后端分离的架构，同时 url 路径采用相对路径，保证了数据的安全性。采用封装良好，成熟的框架进行前前后端框架的搭建，保证平台的稳定性和安全性，同时便于维护。

## 四. 开发流程

### 4.1 开发设计流程图

下图是本次实习的开发流程图，采用前后端分离的架构，通过接口进行访问。

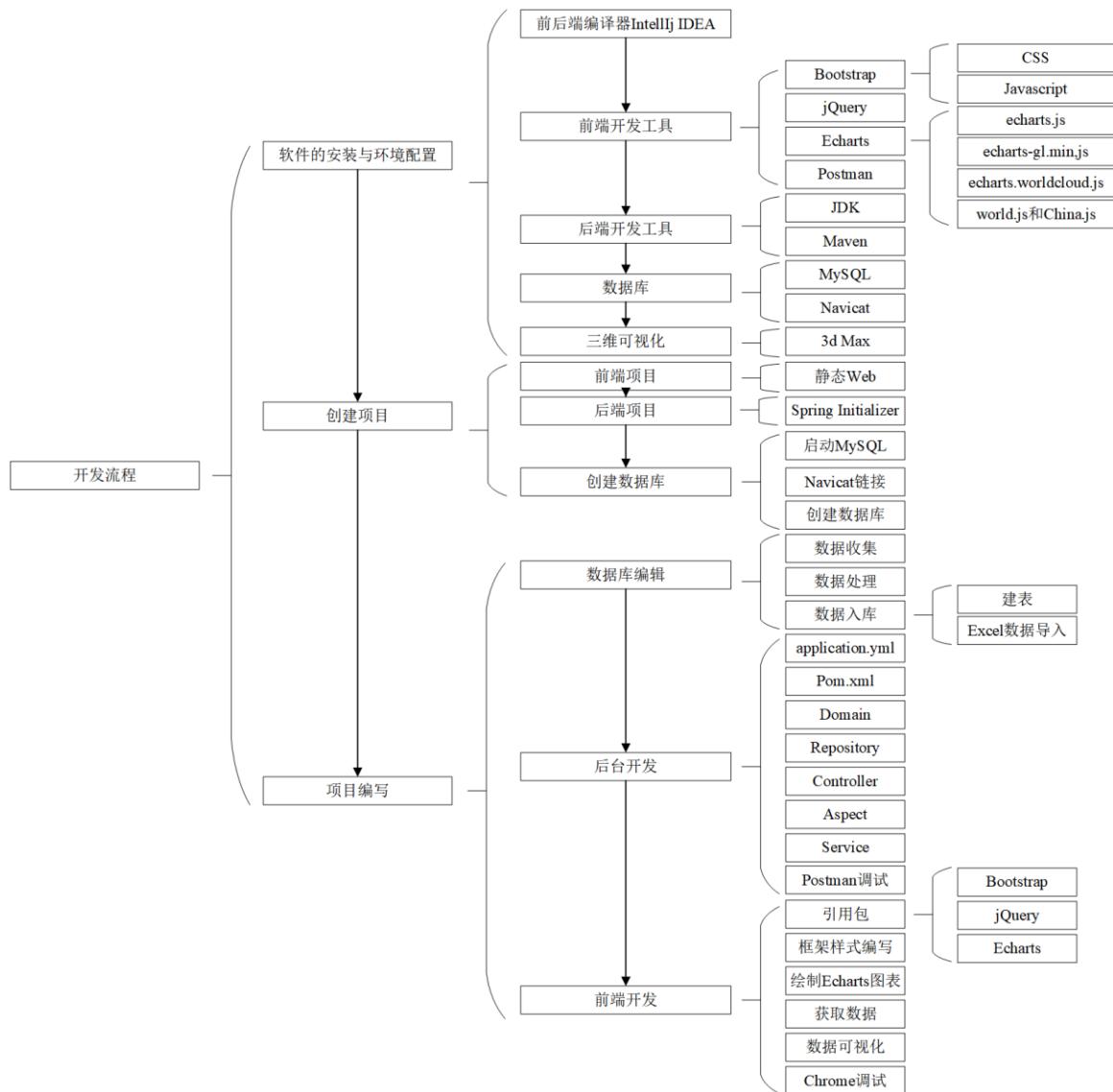


图 4.1 开发流程图

## 4.2 软件安装与环境配置

### 4.2.1 前后端编译器的安装

<https://www.cnblogs.com/123hqb--/p/7552022.html>

1. 官网下载 IntelliJ IDEA 安装包，下载 Ultimate 版

<https://www.jetbrains.com/>

2. 双击安装包，按默认步骤安装

3. 在线激活：点击 Activate，输入：

<http://idea.iteblog.com/key.php>

### 4.2.2 前端的开发工具

1. Bootstrap：从模板网站下载，解压即可

<http://www.cssmoban.com/tags.asp?page=3&n=bootstrap>

2. jQuery：从官网下载 jQuery 库，下载 Development version，用于测试和开发（未压缩，是可读的代码）。解压即可

<https://jquery.com/>

3. Echarts：从官网下载 JavaScript 插件。下载解压即可

- 基本插件 echarts.js，用于绘制基本图表
- WebGL 扩展包 echarts-gl.min.js，提供了丰富的三维可视化组件以及常规图表性能增强，用于绘制三维地球
- 扩展包 echarts.worldcloud.js，用于绘制关键词云
- 扩展包 world.js 和 China.js，用于绘制中国地图和全球地图

<https://www.echartsjs.com/zh/download.html>

4. Postman 安装：

- 官网下载安装包

<https://www.getpostman.com/downloads/>

- 按默认路径安装后，创建账号或用谷歌浏览器账号登录

### 4. 2. 3 后端的开发工具

#### 1. JDK 安装（JDK 中已经有 JRE）：

<https://blog.csdn.net/kangmiao89757/article/details/9993887>

- 从官网下载 64 位 JDK 安装包  
<https://www.oracle.com/technetwork/java/index.html>
- 双击安装包，按默认步骤安装
- 配置环境变量：右击我的电脑→属性→高级→环境变量
  - 点击新建：变量名：JAVA\_HOME，变量值：你的 java 安装所在路径（D:\ProgramFiles\Java\jdk1.7.0\_21）
  - 点击 Path：添加以下变量  
%JAVA\_HOME%\bin; %JAVA\_HOME%\lib\dt.jar; %JAVA\_HOME%\lib\tools.jar
- 检验：在 cmd 中，分别输入 javac 和 java，返回相应内容，则配置安装成功

#### 2. Maven 安装（注意与 JDK 版本的对应）：

[https://blog.csdn.net/github\\_37759996/article/details/90748461](https://blog.csdn.net/github_37759996/article/details/90748461)

- 从官网下载安装包  
<https://maven.apache.org/download.cgi>
- 解压安装包文件到安装目录即可
- 配置环境变量
  - 点击新建：变量名：MAVEN\_HOME，变量值：你的 maven 解压路径（D:\ProgramFiles\Apache\maven）
  - 点击 Path：添加变量%MAVEN\_HOME%\bin;

- 检验：在 cmd 中，输入 maven - version，正确返回版本号等信息则安装成功

PS：由于 Spring Boot 已经集成 Tomcat，不需要再次单独安装

#### 4. 2. 4 数据库的安装配置

##### 1. MySQL 安装：

<https://www.runoob.com/w3cnote/windows10-mysql-installer.html>

- 从官网下载安装包，选择 Windows 类型，离线安装版本  
<https://dev.mysql.com/downloads/mysql/>
- 双击安装包，按默认步骤安装，设置服务器端口和密码
- 配置环境变量：
  - 点击新建：变量名：MYSQL\_HOME，变量值：你的 maven 解压路径（D:\ProgramFiles\MySQL\MySQL Server 5.7）
  - 点击 Path：添加变量% MYSQL\_HOME %\bin;
- 检验：在 cmd 中输入 mysql - u root - p，输入密码，即可操作

##### 2. Navicat 安装：

<https://www.jianshu.com/p/f277c981ec46>

- 下载破解版安装包  
<http://youzfx.cn/resource/1>
- 双击安装包，按默认步骤安装

#### 4. 2. 5 三维可视化软件的安装

##### 1. 3d Max 安装

[https://mp.weixin.qq.com/s/KSSp\\_zLYxvQ-INUQu7uFnA](https://mp.weixin.qq.com/s/KSSp_zLYxvQ-INUQu7uFnA)

- 下载安装包
- 按默认步骤安装
- 输入序列号，激活软件

## 4.3 创建项目

### 4.3.1 前端项目

1. 打开 IntelliJ IDEA，选择创建项目
2. 选择创建静态 Web 项目
3. 选择项目存放地址并命名
4. 新建一个 HTML 页面
5. 将 Bootstrap 模板的库文件，jQuery 包，Echarts 包均与 HTML 文件放在同一目录下

### 4.3.2 后端项目

1. 打开 IntelliJ IDEA，选择创建项目
2. 选择创建 Spring Initializer 项目
3. Project SDK 选择之前下载安装好的 JDK
4. Choose Initializer Service URL 选择  
Default:<http://start.spring.io>
5. 填写 Group 和 Artifact 信息，注意 Artifact 不能包含大写字母，其他按照默认即可  
Type : Maven Project  
Packaging : 项目打包方式，可以选择 jar 包或 war 包，由于 spring-boot 已集成 tomcat，故可以用 java -jar 方式运行
6. 选择主体功能，Web，选择 Spring Boot 版本 2.1.3

7. 修改工程名和路径选择
8. 进入项目，将.mvn、mvnw、mvnw.cmd 删除
9. 查看创建的项目信息：

```
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>2.1.3.RELEASE</version>
  <relativePath/> <!-- lookup parent from repository -->
</parent>
```

图 4.2 Spring Boot 版本

```
<properties>
  <java.version>1.8</java.version>
</properties>
```

图 4.3 Java 版本

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

图 4.4 Web 后台项目

```
<build>
  <plugins>
    <plugin>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-maven-plugin</artifactId>
    </plugin>
  </plugins>
</build>
```

图 4.5 Maven 打包的插件

### 4.3.3 创建数据库

1. 在服务中启动 MySQL
2. Navicat for MySQL 链接数据库
  - 点击文件->新建连接->MySQL
  - 输入密码，查看端口号是否正确
  - 确定，链接成功

3. 创建数据库，设置数据库名，字符集选择 utf8mb4 -- UTF-8 Unicode

## 4.4 项目编写

### 4.4.1 数据库编辑

#### 1. 数据收集

美食文化数据: <https://www.meishi.j.net/>

地图数据: <https://www.echartsjs.com/zh/index.html>

气象数据: <http://data.cma.cn/>

其它属性数据，通过多种来源进行收集，包括新闻网站、博客、论坛等

#### 2. 数据处理

对获取的数据进行筛选和整理，根据需要，选取所需的数据摘录到 Excel 表中。其中图片存储的是相对地址

#### 3. 数据入库

➤ 根据功能设计中的需求，建立若干数据库的表

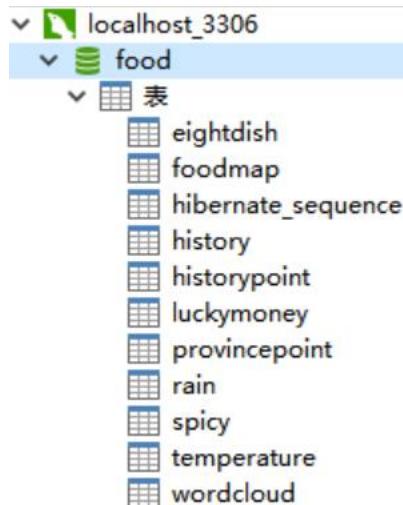


图 4.6 数据库结构图

➤ 设置每一个表的属性字段名和数据类型等，设置主键，注意属性名

与数据处理中的Excel表第一行列的名称对应。选择引擎为InnoDB  
 (该引擎可以支持服务) 如以下示例:

id	province_name	longitude	latitude
1	安徽	117.29	32.0581
2	北京市	116.4551	40.2539
3	福建	119.4543	25.9222
4	甘肃	103.5901	36.3043

图 4.7 Excel 表

名	类型	长度	小数点	不是 null
<b>id</b>	int	2	0	<input checked="" type="checkbox"/> 1
province_name	varchar	255	0	<input checked="" type="checkbox"/>
longitude	decimal	20	4	<input checked="" type="checkbox"/>
latitude	decimal	20	4	<input checked="" type="checkbox"/>

图 4.8 设计属性表

引擎:	InnoDB
字符集:	utf8mb4 -- UTF-8 Unicode
排序规则:	utf8mb4_general_ci
自动递增:	
行格式:	DYNAMIC

图 4.9 更改引擎

- 将 Excel 表另存为 csv, 逗号分隔, utf-8 编码
- 选择导入数据, 导入 csv 表

id	province_name	longitude	latitude
1	安徽	117.2900	32.0581
2	北京	116.4551	40.2539
3	福建	119.4543	25.9222
4	甘肃	103.5901	36.3043

图 4.10 数据库中的表 (与 Excel 是对应的)

## 4.4.2 后台开发

### 1. 后端总体框架

通过使用 Spring Boot 框架, 搭建后端。总体框架如下:

pom.xml: Maven 的配置文件

DemoApplication: 项目启动文件

Application.yml: 项目配置文件，设置端口号等，连接数据库  
Controller: 前端与后台的接口 url，前端 ajax 与 controller 交互  
Domain: 对象，与数据库中的表对应  
Repository: 对象与数据库的接口，继承 jpa，实现数据库操作  
Service: 服务  
Aspect: AOP 分片

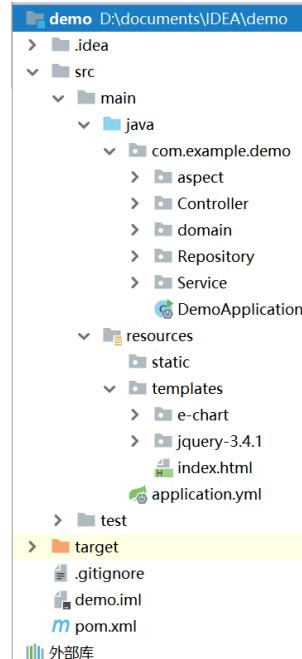


图 4.11 后端总体架构

## 2. 后端详细开发

- 1) 在项目配置文件 application.yml 中编写项目的基本配置，设置端口号，路径等信息

```
server:  
  port: 8081  
servlet:  
  | context-path: /demo
```

图 4.12 项目基本配置

- 2) 数据库操作依赖：使用 Spring-Data-Jpa 实现后台对数据库的操作。实现不用写 SQL 语句却能完成对数据库的操作。在 pom.xml 中，引入 JPA 和 MySQL 的依赖（不用添加 MySQL 版本号）。

从新导入 pom.xml (每一次修改 Maven 配置文件都需要重新导入 pom.xml )

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
</dependency>
```

图 4.13 添加依赖

- 3) 在项目配置文件 application.yml 中编写数据库的配置，在 url 中写入数据库的端口号，数据库的名称，配置时区，用户名和密码，设置数据库的访问方法： update

```
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://localhost:3306/food?useUnicode=true&characterEncoding=utf-8
    username: root
    password: jyh415

  jpa:
    hibernate:
      ddl-auto: update
      show-sql: true
    mvc:
      view:
        prefix: /WEB-INF/
        suffix: .jsp
```

图 4.14 数据库配置

- 4) 建立对象，设置属性和方法：后台中的对象，对应于数据库中的表，对象的属性即为表的属性字段，对象的方法为获取属性值和修改属性。

数据库中的独有的表为 hibernate\_sequence, 为后台控制数据库的 id 序号（若插入数据，序号依次递增，反之则反）

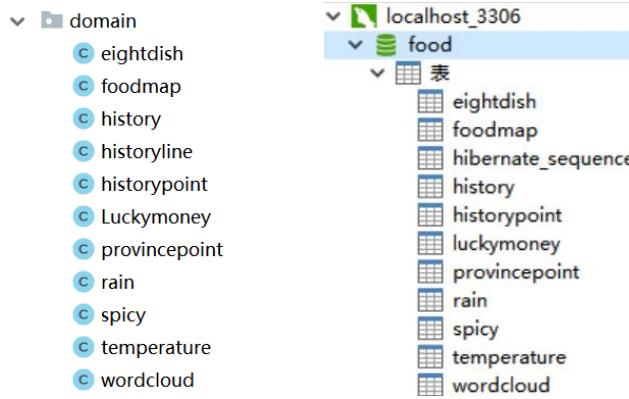


图 4.15 对象总和

```

public class eightdish {

    @Id
    @GeneratedValue

    private Integer id;

    private String province_name;

    private String dish_url;

    private String dish_info;

    public eightdish() {
    }

    public Integer getId() { return id; }

    public void setId(Integer id) { this.id = id; }

    public String getProvince_name() { return province_name; }

    public void setProvince_name(String province_name) { this.province_name = province_name; }

    public String getDish_url() { return dish_url; }

    public void setDish_url(String dish_url) { this.dish_url = dish_url; }

    public String getDish_info() { return dish_info; }

    public void setDish_info(String dish_info) { this.dish_info = dish_info; }
}

```

图 4.16 后端对象及其属性方法

- 5) 建立每一个对象与数据库对应的接口，通过继承 JPA 的方法实现对数据库的操作，增、删、改、查

```

public interface eightdishRepository extends JpaRepository<eightdish, Integer> {
}

```

图 4.17 对象与数据库的接口及 Jpa 方法的继承

- 6) 提供前端访问接口：编写 url，为前端访问数据提供接口。使用类的接口获取数据库的数据

```

@RestController
@RequestMapping("/eightdish")
public class eightdishController {

    @Autowired
    private eightdishRepository repository;

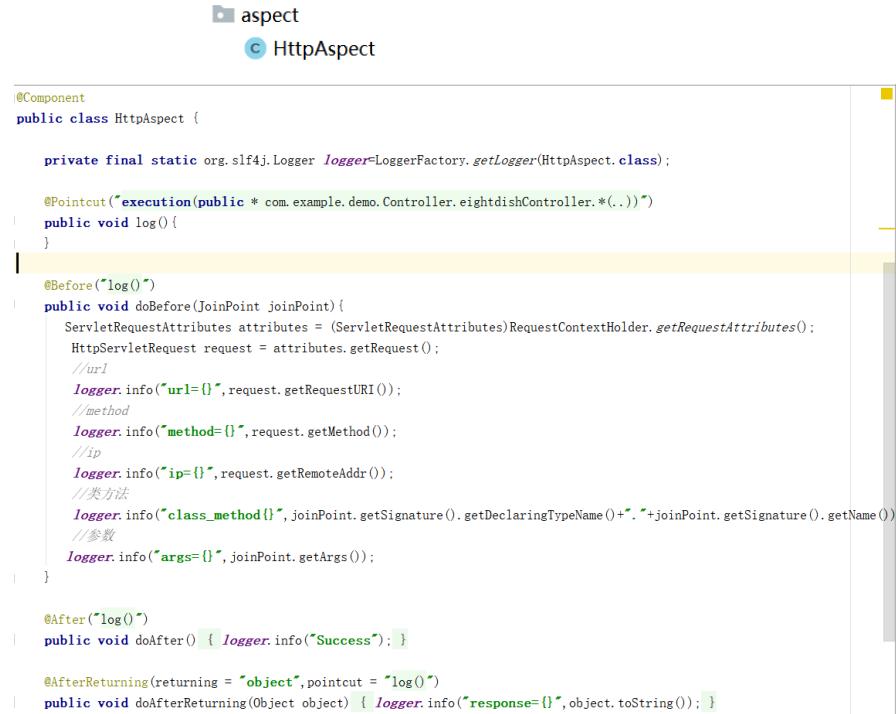
    @CrossOrigin(origins = "*")
    @GetMapping("/data")
    public List<eightdish> list() { return repository.findAll(); }
}

```

图 4.18 前端访问接口及使用 Jpa 方法操纵数据库

- 7) AOP（面向切片编程）：抽取 Controller 的统一方法的切片，比如每次 url 访问时进行用户验证，编写访问日志，便于系统管理。其中 doBefore 方法定义，接口访问前进行的操作，doAfter

定义接口访问成功的操作，doAfterReturning 定义 url 返回的内容。此处定义的是接口访问前后输出接口访问的信息。



```

@Aspect
public class HttpAspect {

    @Component
    public class HttpAspect {
        private final static org.slf4j.Logger logger=LoggerFactory.getLogger(HttpAspect.class);

        @Pointcut("execution(public * com.example.demo.Controller.eightdishController.*(..))")
        public void log() {
        }

        @Before("log()")
        public void doBefore(JoinPoint joinPoint) {
            ServletRequestAttributes attributes = (ServletRequestAttributes)RequestContextHolder.getRequestAttributes();
            HttpServletRequest request = attributes.getRequest();
            //url
            logger.info("url={}",request.getRequestURI());
            //method
            logger.info("method={}",request.getMethod());
            //ip
            logger.info("ip={}",request.getRemoteAddr());
            //类方法
            logger.info("class_method{}",joinPoint.getSignature().getDeclaringTypeName()+"."+joinPoint.getSignature().getName());
            //参数
            logger.info("args={}",joinPoint.getArgs());
        }

        @After("log()")
        public void doAfter() { logger.info("Success"); }

        @AfterReturning(returning = "object", pointcut = "log()")
        public void doAfterReturning(Object object) { logger.info("response={}",object.toString()); }
    }
}

```

图 4.19 AOP 切片

8) 端口调试：在 Postman 中输入端口号，设置请求方法，判断前端是否能正常通过接口请求数据，后台是否能正确的操作数据库，查看 url 返回的数据格式（JSON）

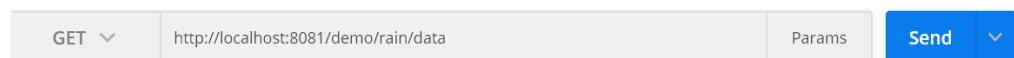


图 4.20 使用 Postman 调试接口

2019-11-25 15:41:23.241 INFO 11248 --- [nio-8081-exec-1] com.example.demo.aspect.HttpAspect : url=/demo/rain/data
2019-11-25 15:41:23.241 INFO 11248 --- [nio-8081-exec-1] com.example.demo.aspect.HttpAspect : method=GET
2019-11-25 15:41:23.242 INFO 11248 --- [nio-8081-exec-1] com.example.demo.aspect.HttpAspect : ip=0:0:0:0:0:0:1
2019-11-25 15:41:23.243 INFO 11248 --- [nio-8081-exec-1] com.example.demo.aspect.HttpAspect : class_methodcom.example.demo.Controller.rainController.list
2019-11-25 15:41:23.243 INFO 11248 --- [nio-8081-exec-1] com.example.demo.aspect.HttpAspect : args={}
2019-11-25 15:41:23.283 INFO 11248 --- [nio-8081-exec-1] o.h.h.i.QueryTranslatorFactoryInitiator : HHH0000397: Using ASTQueryTranslatorFactory
Hibernate: select rain0_.id as id1_6_, rain0_.rchna as rchna2_6_, rain0_.rdate as rdate3_6_, rain0_.rsichuan as rsichuan4_6_ from rain rain0_
2019-11-25 15:41:23.513 INFO 11248 --- [nio-8081-exec-1] com.example.demo.aspect.HttpAspect : Success
2019-11-25 15:41:23.513 INFO 11248 --- [nio-8081-exec-1] com.example.demo.aspect.HttpAspect : response=[rain{id=1, rdate='2010-1', rsichuan=55.5, rchina=63.}]

图 4.21 后台输出

三块后台输出内容分别为：

- doBefore 定义的输出前台访问 url，访问方法等
- 操作数据库的 SQL 语句
- doAfter 和 doAfterReturning 访问成功后的返回内容

```

1 [ [ { "id": 1, "rdate": "2010-1", "rsichuan": 55.5, "rchina": 63.25862069 }, { "id": 2, "rdate": "2010-2", "rsichuan": 52.5, "rchina": 65.63793103 }, { "id": 3, "rdate": "2010-3", "rsichuan": 62.5, "rchina": 61.22413793 } ] ]

```

图 4.22 前端获取的 JSON 数据

#### 4.4.3 前端开发

- 在 HTML 中引用上述 BootStrap 模板和 JavaScript 包
- Bootstrap 模板的引用：参考模板中的 HTML 示例，采用<script>标签引入 js 文件，<link>标签引入 css 文件，可以直接复制示例 HTML 的引用，输入包的路径，如下示例：

```
<!-- Fonts and icons -->
<script src="assets/js/plugin/webfont/webfont.min.js"></script>
```

图 4.23 js 文件的引入

```
<!-- CSS Files -->
<link rel="stylesheet" href="assets/css/bootstrap.min.css">
```

图 4.24 css 文件的引入

- jQuery 和 Echarts 的引用采用<script>标签引用，输入包的路径

```
<script src="jquery-3.4.1/jquery-3.4.1.js"></script>
```

图 4.25 jQuery 的引入

```
<script src="e-chart/echarts.js"></script>
<script src="e-chart/echarts-wordcloud.js"></script>
<script src="e-chart/china.js"></script>
<script src="e-chart/echarts-gl.min.js"></script>
<script src="e-chart/world.js"></script>
```

图 4.26 Echarts 包的引入

2. 样式框架编写：参考 BootStrap 模板的示例 HTML，布局项目的 Web 前端样式框架，在适当的位置，使用<div id=”XXX”>标签留出容器的位置，用于存放绘制 Echarts 图表，不同 id 对应不同容器，至此 Web 前端样式的框架搭建完成

```
<div id="chart1" style="width: 650px; height:410px; margin: 0 auto; "></div>
```

3. 绘制 Echarts 图表：在< Script >标签，使用 GetElementbyID 方法，输入上一步中设置的<div>标签的 id 号，获取对应的用来存放 Echarts 图表的容器。然后根据官网 Echarts 示例，绘制 Echart 图表

```
var myChart = echarts.init(document.getElementById('chart1'));
```

4. 获取数据：使用 ajax 方法获取后端 url 接口的数据，访问成功则将数据储存在变量中，注意此处获取的数据的格式为 JSON，储存数据的变量针对不同的 Echarts 图表所需的数据格式不同。访问失败则在网页中弹框提示

储存数据的变量

```
var Date1=[];
var China1=[];
var Sichuan1=[];
```

访问方法 GET

```
$.ajax({
```

type: "get",

```
url: "http://localhost:8081/demo/temperature/data",
```

data: {},

```
dataType : "json",
```

async: false,

```
success:function (data) {
```

for(var i=0;i<data.length;i++) {

Date1.push(data[i].tdate);

China1.push(data[i].tchina);

Sichuan1.push(data[i].tsichuan);

}

},

```
error : function(errorMsg) {
```

//请求失败时执行该函数

alert("图表请求数据失败!");

myChart.hideLoading();

}



数据访问的接口 url

数据格式 JSON

访问成功后

将数据储存在变量中

访问失败，弹框提示

图 4.27 Ajax 方法数据获取

```
Date2.push(data[i].rdate);
China2.push(data[i].rchina);
Sichuan2.push(data[i].rsichuan);
```

图 4.28 简单的数组数据格式

```
nums.push({name:data[i].province_name,value:data[i].province_value});
```

图 4.29 自定义的数据类型的数组

```
str = "<h2>" + data[i].state_name + "</h2>" + " +
    "<a href=" + data[i].foodmap_urltwo + ">" + "" + "</a>" +
    index = data[i].id.toString();
document.getElementById(index).innerHTML = str;
```

图 4.30 组合成 HTML 语句的数据格式

localhost:63342 显示

图表请求数据失败!

确定

图 4.31 数据访问失败弹框提示

5. 显示数据：在 Echarts 或 HTML 中实现数据的可视化。如下图，表示从后台获取的数据存储在 nums 变量中，作为 Echarts 的数据，通过 Echarts 实现可视化

```
myChart.setOption(
  {
    series:[
      {
        data:nums
      }
    ]
  }
)
```

图 4.32 Echarts 设置数据

6. 在 Chrome 浏览器中调试前端：由于数据格式的编写错误，插件引入错误等的原因，数据获取和显示可能不成功，使用 Chrome 浏览器自带的工具，进行调试，在 watch 可以查看 JavaScript 中变量的值

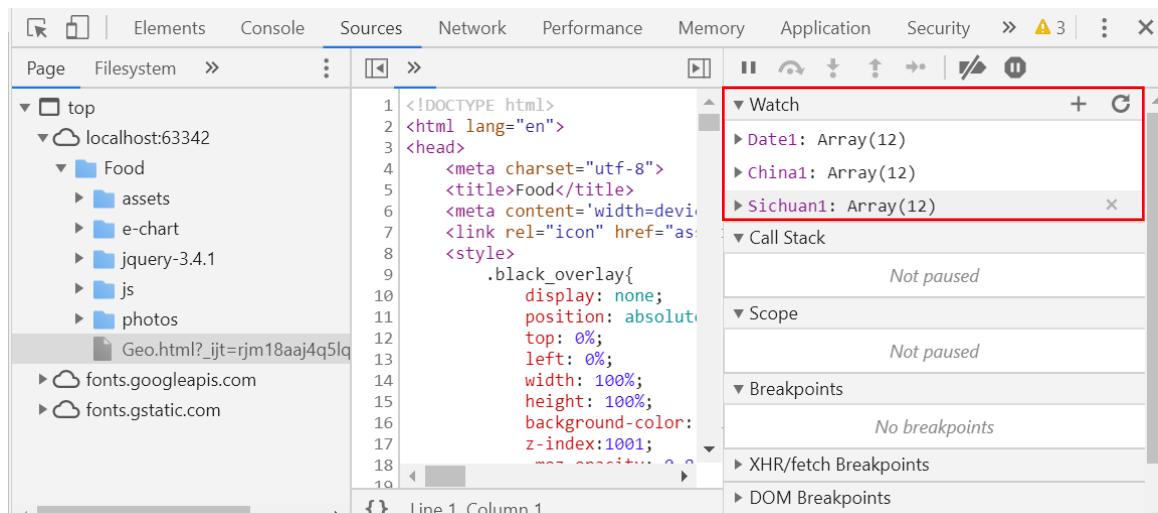


图 4.33 调试窗口

```
▼ Date1: Array(12)
  0: "2010-1"
  1: "2010-2"
  2: "2010-3"
  3: "2010-4"
  4: "2010-5"
  5: "2010-6"
  6: "2010-7"
  7: "2010-8"
  8: "2010-9"
  9: "2010-10"
  10: "2010-11"
  11: "2010-12"
  length: 12
  ▶ __proto__: Array(0)
▶ China1: Array(12)
```

图 4.34 Watch 窗口中查看变量的值以及确认格式是否正确

## 五. 软件介绍

### 5.1 功能点

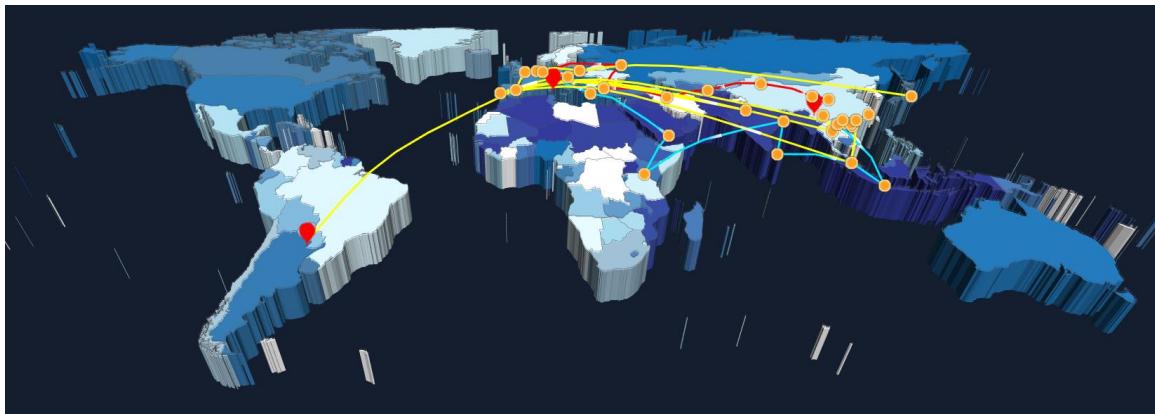
#### 5.1.1 二维地图



### 5.1.2 二维专题地图



### 5.1.3 三维地图



### 5.1.3 路线地图



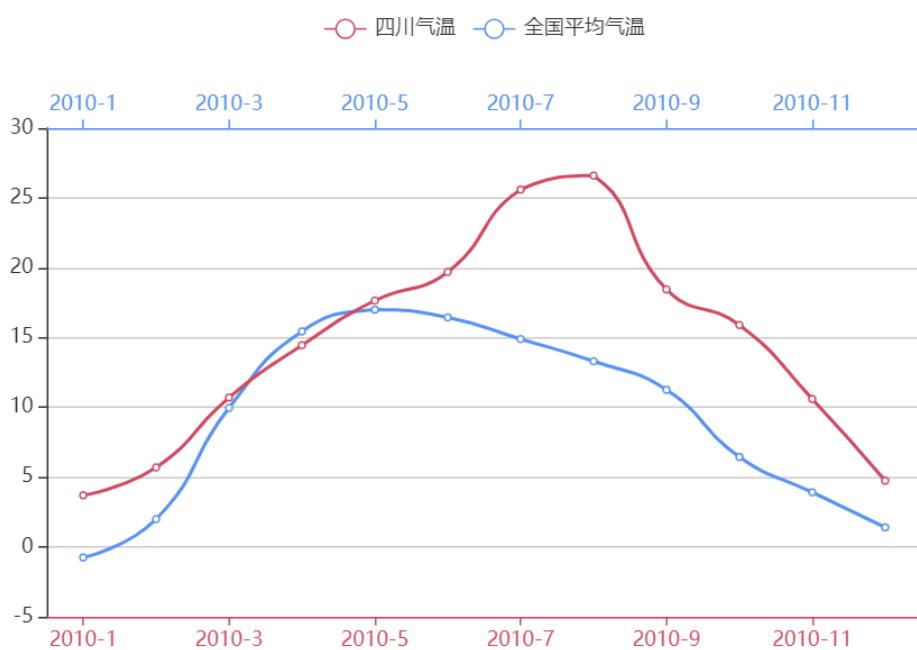
### 5.1.4 三维地球



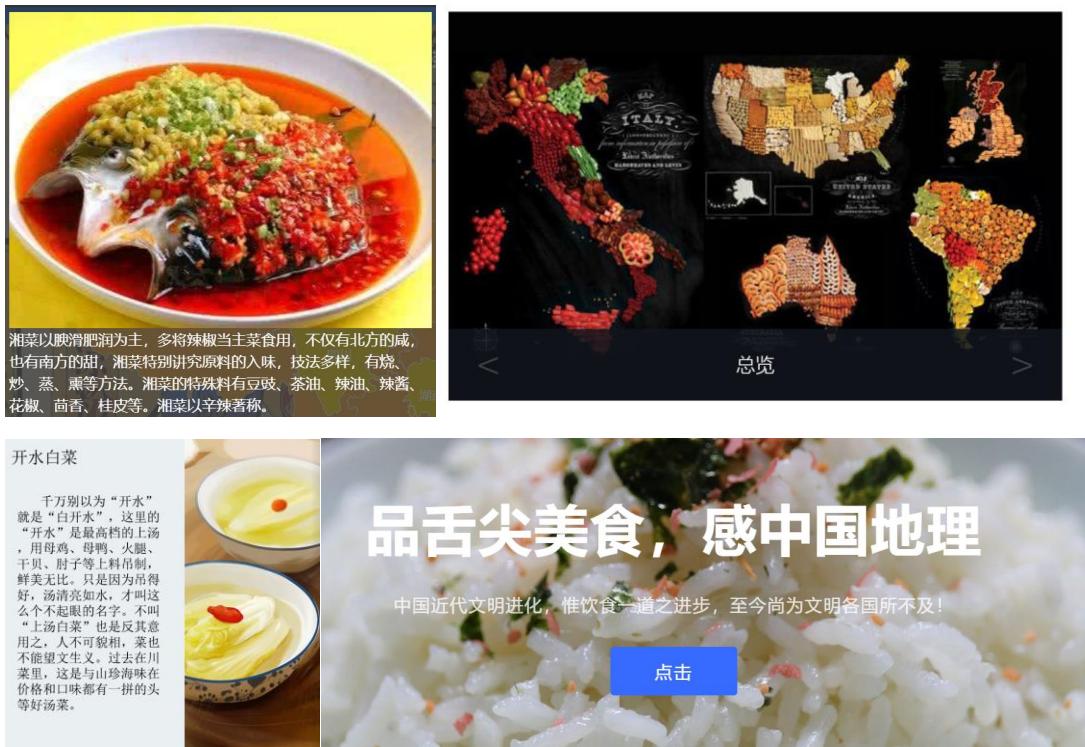
### 5.1.5 词云



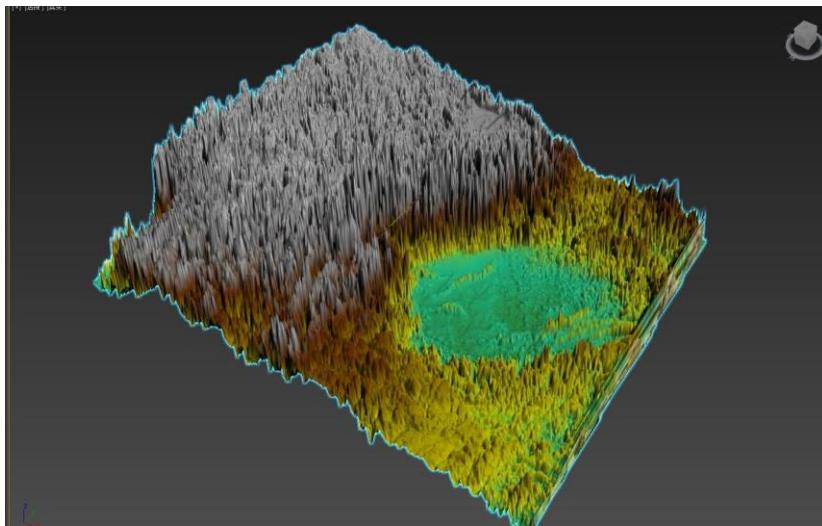
### 5.1.6 折线图



### 5.1.7 图片和文字展示



### 5.1.8 三维模型



## 5.2 基本操作

### 1. 点击按钮进入系统



### 2. 点击按钮或页角，进行翻页查看



### 3. 工具栏点击地图跳转至八大菜系地图



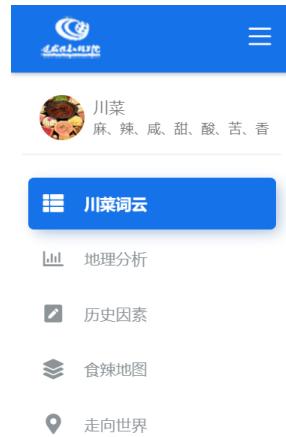
### 4. 鼠标滑过对应省份或文字，浮框显示图片和文字



### 5. 搜索栏输入菜系名，搜索详细介绍



6. 工具栏点击相应分析，查看内容



7. 点击打开或折叠工具栏



8. 点击返回八大菜系主界面



9. 点击右边页脚文字，查看隐藏内容



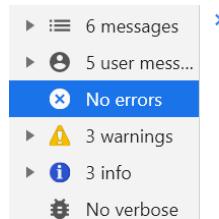
## 5.3 程序运行效果

### 5.3.1 前端运行效果

1. 正确请求数据，无数据请求失败弹框

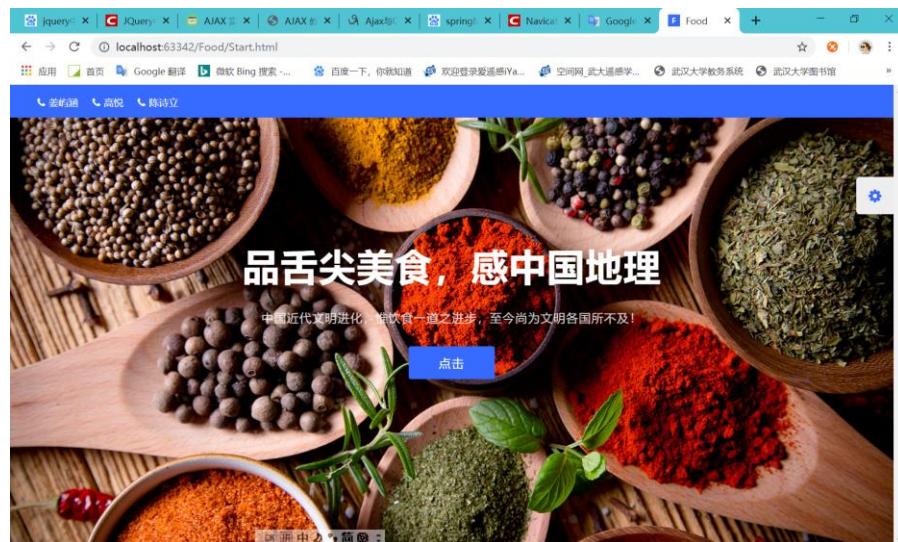
```
▼ Watch
  ▼ Date1: Array(12)
    0: "2010-1"
    1: "2010-2"
    2: "2010-3"
    3: "2010-4"
    4: "2010-5"
    5: "2010-6"
    6: "2010-7"
    7: "2010-8"
    8: "2010-9"
    9: "2010-10"
    10: "2010-11"
    11: "2010-12"
    length: 12
```

## 2. 使用 Chrome 浏览器调试，无错误

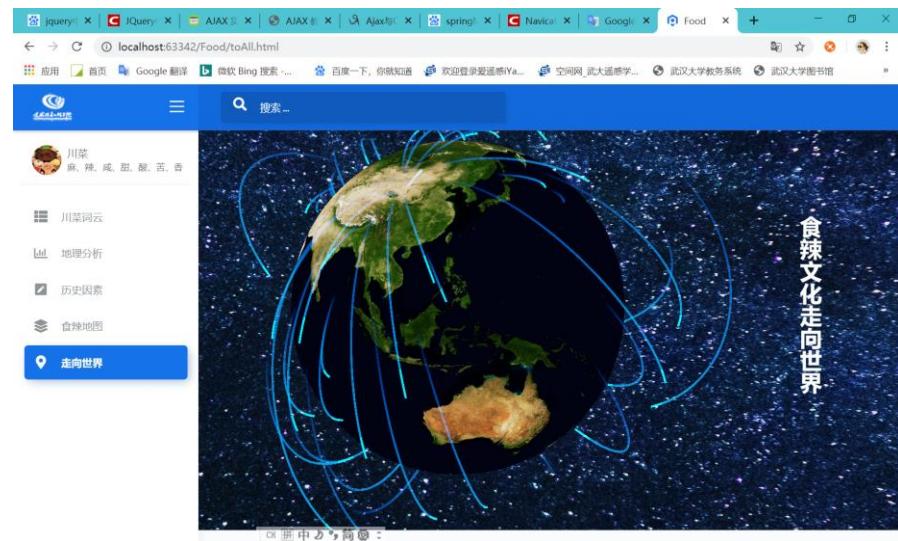


## 3. 页面正常显示

首页：



功能页：



## 5.3.2 后台运行效果

### 1. 后台正常启动

```
2019-11-25 20:29:02.741 INFO 22136 --- [ restartedMain] com.example.demo.DemoApplication : Started DemoApplication in 4.435 seconds (JVM running for 5.53)
```

### 2. 正常完成多次数据库操作，接收 url 请求，返回 JSON 格式数据

```
2019-11-25 19:09:14.421 INFO 22784 --- [nio-8081-exec-2] com.example.demo.aspect.HttpAspect : url=/demo/rain/data
2019-11-25 19:09:14.421 INFO 22784 --- [nio-8081-exec-2] com.example.demo.aspect.HttpAspect : method=GET
2019-11-25 19:09:14.421 INFO 22784 --- [nio-8081-exec-2] com.example.demo.aspect.HttpAspect : ip=0:0:0:0:0:0:0:1
2019-11-25 19:09:14.421 INFO 22784 --- [nio-8081-exec-2] com.example.demo.aspect.HttpAspect : class_methodcom.example.demo.Controller.rainController.list
2019-11-25 19:09:14.421 INFO 22784 --- [nio-8081-exec-2] com.example.demo.aspect.HttpAspect : args={}
Hibernate: select rain0_.id as id1_6_, rain0_.rchna as rchna2_6_, rain0_.rdate as rdate3_6_, rain0_.rsichuan as rsichuan4_6_ from rain rain0_
2019-11-25 19:09:14.425 INFO 22784 --- [nio-8081-exec-2] com.example.demo.aspect.HttpAspect : Success
2019-11-25 19:09:14.426 INFO 22784 --- [nio-8081-exec-2] com.example.demo.aspect.HttpAspect : response=[rain{id=1, rdate='2010-1', rsichuan=55.5, rchna=63.25862069}, rain{
Hibernate: select spicy0_.id as id1_7_, spicy0_.province_name as province2_7_, spicy0_.province_value as province3_7_ from spicy spicy0_
Hibernate: select provincepo0_.id as id1_5_, provincepo0_.latitude as latitude2_5_, provincepo0_.longitude as longitud3_5_, provincepo0_.province_name as province4_5_ from provinc
```

## 5.4 关键代码

### 5.4.1 前端关键代码

#### 1. 引入 BootStrap, jQuery, Echarts

##### ➤ BootStrap

```
<link rel="icon" href="assets/img/icon.ico" type="image/x-icon"/>

<!-- Fonts and icons --&gt;
&lt;script src="assets/js/plugin/webfont/webfont.min.js"&gt;&lt;/script&gt;
&lt;script&gt;
    WebFont.load({
        google: {"families":["Lato:300,400,700,900"]},
        custom: {"families":["Flaticon", "Font Awesome 5 Solid", "Font Awesome 5 Regular", "Font Awesome 5 Brands", "simple-line-icons"], "active": function() {
            sessionStorage.fonts = true;
        }}
    });
&lt;/script&gt;

<!-- CSS Files --&gt;
&lt;link rel="stylesheet" href="assets/css/bootstrap.min.css"&gt;
&lt;link rel="stylesheet" href="assets/css/atlantis.min.css"&gt;</pre>
```

```

<!-- Core JS Files -->
<script src="assets/js/core/jquery.3.2.1.min.js"></script>
<script src="assets/js/core/popper.min.js"></script>
<script src="assets/js/core/bootstrap.min.js"></script>
<!-- jQuery UI -->
<script src="assets/js/plugin/jquery-ui-1.12.1.custom/jquery-ui.min.js"></script>
<script src="assets/js/plugin/jquery-ui-touch-punch/jquery.ui.touch-punch.min.js"></script>
<!-- jQuery Scrollbar -->
<script src="assets/js/plugin/jquery-scrollbar/jquery.scrollbar.min.js"></script>
<!-- Chart JS -->
<script src="assets/js/plugin/chart.js/chart.min.js"></script>
<!-- jQuery Sparkline -->
<script src="assets/js/plugin/jquery.sparkline/jquery.sparkline.min.js"></script>
<!-- Chart Circle -->
<script src="assets/js/plugin/chart-circle/circles.min.js"></script>
<!-- Datatables -->
<script src="assets/js/plugin/datatables/datatables.min.js"></script>
<!-- Bootstrap Notify -->
<script src="assets/js/plugin/bootstrap-notify/bootstrap-notify.min.js"></script>
<!-- jQuery Vector Maps -->
<script src="assets/js/plugin/jqvmap/jquery.vmap.min.js"></script>
<script src="assets/js/plugin/jqvmap/maps/jquery.vmap.world.js"></script>
<!-- Sweet Alert -->
<script src="assets/js/plugin/sweetalert/sweetalert.min.js"></script>

<!-- Atlantis JS -->
<script src="assets/js/atlantis.min.js"></script>

```

## ➤ jQuery

```
<script src="jquery-3.4.1/jquery-3.4.1.js"></script>
```

## ➤ Echarts

```

<script type="text/javascript" src="e-chart/echarts.js"></script>
<script type="text/javascript" src="e-chart/echarts-wordcloud.js"></script>
<script type="text/javascript" src="e-chart/china.js"></script>
<script type="text/javascript" src="e-chart/echarts-gl.min.js"></script>
<script type="text/javascript" src="e-chart/data-1556466081698-qDedSQsRX.js"></script>
<script type="text/javascript" src="e-chart/world.js"></script>

```

## 2. Echarts 绘图

```

<script >
    var myChart = echarts.init(document.getElementById('main'));
    var $imgs = [];
    var value=[];
    $.ajax({ "type": "get" ... });

    var option = {"backgroundColor": '#013954' ...};

    myChart.setOption(option);

    window.onresize = myChart.resize;
</script>

```

## 3. AJAX 数据请求

```

$.ajax({
    type:"get",
    url:"http://localhost:8081/demo/wordcloud/data",
    data:{},
    dataType : "json",
    async: false,
    success:function (data) {
        for(var i=0;i<data.length;i++) {
            $imgs.push({area:data[i].keyword,url:data[i].word_url});
            value.push({name:data[i].keyword,value:data[i].value})
        }
    },
    error : function(errorMsg) {
        //请求失败时执行该函数
        alert("图表请求数据失败!");
        myChart.hideLoading();
    }
});

);

```

## 5.4.2 后端关键代码

### 1. 数据库配置

```

spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://localhost:3306/food?useUnicode
    username: root
    password: jyh415

  jpa:
    hibernate:
      ddl-auto: update
      show-sql: true
  mvc:
    view:
      prefix: /WEB-INF/
      suffix: .jsp

```

### 2. 创建对象

```

@Entity
public class eightdish {
    @Id
    @GeneratedValue
    private Integer id;
    private String province_name;
    private String dish_url;
    private String dish_info;
    public eightdish() {}
    public Integer getId() { return id; }
    public void setId(Integer id) { this.id = id; }
    public String getProvince_name() { return province_name; }
    public void setProvince_name(String province_name) { this.province_name = province_name; }
    public String getDish_url() { return dish_url; }
    public void setDish_url(String dish_url) { this.dish_url = dish_url; }
    public String getDish_info() { return dish_info; }
    public void setDish_info(String dish_info) { this.dish_info = dish_info; }
    @Override
    public String toString() {...}
}

```

### 3. 数据库操作接口，继承 JPA 方法

```
public interface eightdishRepository extends JpaRepository<eightdish, Integer> {  
}
```

### 4. Controller 编写

```
@RestController  
@RequestMapping("/eightdish")  
public class eightdishController {  
  
    @Autowired  
    private eightdishRepository repository;  
  
    @CrossOrigin(origins = "*")  
    @GetMapping("/data")  
    public List<eightdish> list() { return repository.findAll(); }  
}
```

### 5. AOP 分片

```
@Aspect  
@Component  
public class HttpAspect {  
  
    private final static org.slf4j.Logger logger= LoggerFactory.getLogger(HttpAspect.class);  
  
    @Pointcut("execution(public * com.example.demo.Controller.rainController.*(..))")  
    public void log(){  
    }  
  
    @Before("log()")  
    public void doBefore(JoinPoint joinPoint){...}  
  
    @After("log()")  
    public void doAfter(){ logger.info("Success"); }  
  
    @AfterReturning(returning = "object", pointcut = "log()")  
    public void doAfterReturning(Object object) { logger.info("response={} ", object.toString()); }  
}
```

## 六. 总结

在本次实习中，我主要负责系统的开发，虽然之前有一点 Web 前端开发的经验，但能力的确不足，在开发过程中，很多需求都不能满足，在不断的学习过程中，从一开始几乎全部静态的系统，到如今拥有各种交互能力的系统，作品经过了无数次的打磨。而在后台方面，以前一直是我的盲区，不论如何都把后台搭建不

好，不能使用前端调用数据。而在本次的开发过程中，从建立数据库，到搭建后台，实现前端访问后台数据，我成功的跨过了这一道坎，真正实现了完成一个完整的系统。收获颇丰。

## 6.1 关于系统本身

在前期走了很多的弯路，从选题开始，一直找不到立意好的主题，而我们自己感兴趣的主題又很难做出亮点和特色。最终确定做一个我们感兴趣的主題美食。那么美食要做什么呢，一开始，我们按照论文，使用 ArcGIS 进行美食路线的规划，然而这个主题却显得太过单调和普通。后来通过查看大量的文献论文，听取老师的意见后才最终在美食上确定了较好的立意，从多个角度多个层次，去探究美食文化与地理、民俗的关系。拥有好的立意就成功了一半，很多时候不是系统功能做的不好，而是系统本身的适用性不得人心。在做 GIS 软件开发时，一定要学会观察和思考，前期要多阅读多参考，扩宽知识面，才能整合出好的电子。

在系统实现过程中，我体会到开发软件，是一个不断打磨，翻来覆去改进的过程。从静态网页到有用户交互的动态网页，到拥有完整后台的完整软件。当不断的有新的想法，新的需求时，就需要往系统中不断加入新的功能，不断琢磨如何让系统能够更好的阐释主题，更好的“讲故事”，把美食文化完美的呈现出来。一直不明白前端如何获取后台的数据，而后台又如何访问数据库，不知道相互分离的三大块，如何神奇的实现连接。后来结合相应的教程，通过向学长请教等方式，总算走通了这条路。当前端第一次成功访问到数据时，内心无比激动，像魔法一般。在往后的学习中，也要这样一步步的走，总会有意想不到的收获，即使有一道大坎，不断的学习，总能突破，这是本次课程带给我最大的收获。

## 6.2 关于小组合作

非常幸运，本次我们小组三人都很靠谱，分工明确，积极工作，各有所长，各司其职，当遇到问题时也不气馁，大家一起想办法解决。

小组的氛围十分的重要。我们小组的氛围一直都很轻松，有时间大家就约在一起做，也一起学习。和谐轻松的氛围，使得大家都能够随意的发表自己的想法，三个人的想法凝聚在作品上，才有了如今这样功能丰富充实饱满的系统。

良好的合作和信任是成功的基石。一方面是组长的统领作用，一方面是任务分工的合理。组长的统领，使得各方面的工作能够对接上，安排进度流程保证项目顺利的完成。其次是合作，大家都很认真负责，在本次项目开发过程中，当我在做前端和后端的时候，丝毫不担心数据库会有问题或者数据收集方面有问题。

## 七. 软件特色及改进建议

### 7.1 软件特色

在数据分析方面，软件实现了多角度多层次的分析，由点及面，以四川为例，展示中国丰富的饮食文化，及其背后的地理以及文化的原因。在分析川菜的辣文化过程中，结合 3D 模型、温湿度曲线图、嗜辣专题地图、辣椒传播路线图，从多个角度，阐释川菜中的辣文化，一方面是从自然的角度因为地理环境的潮湿，通过吃辣来除湿，另外一方面，从人文的角度，由丝绸之路将辣椒带入中国。从而说明在全国范围内饮食文化的形成与地理人文密不可分，同时通过一组由食物组成的世界地图，也可以看到饮食文化的形成与地理人文因素密不可分。

在数据可视化方面，一是我们采用了非常形象生动展现形式，在解释地理因素时，以动态雨点的形式，说明四川地区多雨，用三维地形图解释四川盆地的潮湿。另外一方面，我们的可视化形式非常的丰富多样，包括，折线图、三维模型、专题地图、路线图等，非常完美的去将数据展现出来，弥补很多关于饮食分析中，只有文字描述的不足。

在系统构建方面，我们采用了当前 Web 开发非常受欢迎的两大架构，以及工具和编程方式，前端的 Bootstrap 框架、Echarts 图表和后端的 Spring Boot、AOP 面向切片的编程。前后端的架构使得我们的系统更加稳定可靠，开发也更加简洁。

从系统功能上来说，我们的操作界面友好，用户能够直接上手实现软件的使用，按钮，列表等的设计布局也很合理

## 7.2 问题及建议

系统在用户交互方面还可以加强，由于开始的设计过程，没有考虑用户登录这一方面的管理，因此用户管理这一方面还存在欠缺。其次平台主要功能是用于展示自身特点，可以更多的加入一些用户的操作。

在主题方面，目前只实现了在川菜辣文化方面的深入，其余菜系还有待更加深入的挖掘，完善。同时功能上也可以进行更多的扩展，比如各大菜系网红美食地点的推荐，规划美食路线等，增加平台的实用性。

在数据方面，目前的数据主要靠小组成员人工进行收集，没有找到这方面对口的大型数据库，从系统的扩展性来看，希望能够实现一种自动化的数据收集，找到更好的数据资源。