数据可视化中心的开发

摘要

数据可视化是近年来的一个热点话题，也是数据分析的一个关键技术。本文通过建立一个网站用于承载数据可视化的图表和数据。

首先对所用到的技术进行了简要介绍，然后开始编写前端页面和后端逻辑代码，紧接着介绍了服务器和服务器软件的概念以及使用方法， 按照前端、后端、服务器的顺序一步一步的完成了网站的构建，并且搭建了可动态查询的数据库系统，然后对使用到的多种技术进行了介绍，最后指出了本次项目的不足之处，并且为未来更复杂需求的场景提供了代码重构的方向。

在所有的步骤都完成之后，一个美丽、直观、高效的数据可视化网站也呈现在眼前，通过此网站可以方便专业人员对数据进行分析和整理，以便提前做出下一步的判断。

关键字：计算机技术；Web技术；服务器；Django；Echarts；Nginx；uWSGI

Development of Data Visualization Center

Abstract

Data visualization is a hot topic in recent years and a key technology for data analysis. This article builds a website for hosting data visualization charts and data.

First of all, a brief introduction to the technologies used, and then began to write front-end pages and back-end logic code, followed by the concept and usage of server and server software, completed step by step in the order of front-end, back-end, and server Build the website and build a database system that can be dynamically queried, then introduce the various technologies used, and finally point out the shortcomings of this project, and provide code refactoring for future scenarios with more complex requirements direction.

After all the steps are completed, a beautiful, intuitive and efficient data visualization website is also presented. Through this website, it is convenient for professionals to analyze and organize the data, so as to make the next judgment in advance.

Keywords: Computer Technology; Web Technology; Server; Django; Echarts; Nginx; uWSGI

目 录

[1 绪论 1](#_Toc103606577)

[1.1 引言 1](#_Toc103606578)

[1.2 WEB前端技术概述 1](#_Toc103606579)

[1.3 基于Python的Django框架概述 3](#_Toc103606580)

[1.4 本文主要内容 4](#_Toc103606581)

[2 前端页面的结构搭建与美化 6](#_Toc103606582)

[2.1 主界面的设计与布局 6](#_Toc103606583)

[2.2 页面美化 7](#_Toc103606584)

[2.3 图表引入界面 8](#_Toc103606585)

[2.4 中国地图页面引入界面 9](#_Toc103606586)

[2.5 地图点击功能实现 10](#_Toc103606587)

[2.6 二级页面制作 12](#_Toc103606588)

[3 后端配置以及设置路由 14](#_Toc103606589)

[3.1 启动项目 14](#_Toc103606590)

[3.2 路由配置 14](#_Toc103606591)

[3.3 静态资源定位 14](#_Toc103606592)

[3.4 数据库设置 14](#_Toc103606593)

[3.5 数据接口编写 15](#_Toc103606594)

[4 服务器配置 17](#_Toc103606595)

[4.1 云服务器配置 17](#_Toc103606596)

[4.2 MySQL数据库搭建与配置 18](#_Toc103606597)

[4.3 服务器软件的配置 19](#_Toc103606598)

[5 设计方案的影响分析 22](#_Toc103606599)

[结论 23](#_Toc103606600)

[致谢 24](#_Toc103606601)

[参考文献 25](#_Toc103606602)

## 绪论

### 1.1 引言

由于数据本身只是一组数字，它们在普通人眼里并没有很直观的意义，所以数据可视化对于数据的展现就显得十分有必要了。通过各种图表，图示对数据进行全方位的展示可以很方便的让普通民众直观的看到数据的意义以及它的变化趋势，同时能够帮助专业从业人员对数据的变化规律进行分析、判断与筛选，从而帮助做出下一步的决策。可视化技术在近几年来飞速发展，目前市面上有许多优秀的数据可视化工具，其中，由中国百度公司带头编写的Echarts工具不仅在国内得到了广泛的应用，也受到了国际开源组织Apache的青睐，被其写入Apache协议。本次的项目便是围绕着Echarts框架，并且基于WEB前端技术，最终实现一个能在浏览器上浏览的，在线的数据可视化网站。接下来的几个小结将会简要介绍以下本次项目将会使用到的几个技术。

### 1.2 WEB前端技术概述

#### ​1.2.1 HTML与HTML5简介

HTML英文直译为超文本标记语言，虽然听起来很像，但是严格来说，它并不属于一种编程语言。HTML被用来编写网页的结构，它是Web的发明者Tim Berners-Lee和同事 Daniel W. Connolly于1990年创立的一种标记语言[1]。由浏览器将HTML文件“翻译”成可以识别的信息（这个过程称为“渲染”），即所谓的网页。每一个网页都是一个单独的HTML文件，这个文件里面包含了HTML的标签，这些标签是一种用于搭建网页基本结构的语言。HTML的逐渐流行使得超链接越来越多的出现在网站上，也就是通过点击网页上的一个节点从一个网页跳到其它的网页。网页的本质可以说就是一个HTML文件，通过结合其他的WEB技术，如下文将要介绍的CSS和JavaScript技术，就可以让程序员创造出功能强大，样式美观的网页应用。

HTML5是互联网的下一代标准，HTML5再从前的HTML4的基础上进行了一定的改进，它对之前市面上流行的各种浏览器进行了整合适配，能够让消费者从个人电脑、笔记本电脑、智能手机或平板电脑在内的任意终端访问相同的程序和基于云端的信息[2]。互联网管理团队为HTML5新标准使用了本地存储技术，这大大减少了用户的等待时间，优化了用户的体验，目前绝大多数的前端网页页面的结构都已经是由HTML5编写。

#### ​1.2.2 CSS与CSS3简介

CSS英文直译为层叠样式表，它是一种用来为基于HTML或者是XML构建的架构化的网页设置美化效果的计算机语言，由于XML技术已经逐渐淡出在技术世界里，CSS就成为了一种主要与HTML配合使用的关键性技术。CSS的基本功能就是为网页添加各种样式，如颜色，边框，动画等等，它不仅可以静态地修饰网页，还可以配合各种脚本怨言动态的对网页各个元素进行格式化。CSS能够对网页中元素位置的排版进行像素级别的精确控制，几乎支持所有字体字号样式，拥有对网页对象和模型样式编辑的能力[3]。CSS样式可以直接写在HTML网页内部或者单独的一个样式文件，样式文件的扩展名为.css，CSS规则是由一个或者多个属性与其值组成的，也就是键值对形式。CSS名字中的“层叠”表达了每单个规则作用与网页元素的方式，每个样式之间形成一个层次结构，更具体的样式会覆盖更通用的样式，所以称之为“层叠”。

CSS3是CSS的新一代标准，由于网速的提升与浏览器性能的发展，用户对网页的要求越来越高，于是CSS3和HTML5诞生了，CSS3提供了对网页的更多的样式，例如动画等等，并且大幅度优化了性能，使得网页更为流畅。

#### ​1.2.3 JavaScript简介

JavaScript是一种高级解释性编程语言，这一点与Python类似，同样的，JS也是世界上被使用最广泛的编程语言之一，它可以用于前端，后端，系统应用，手机APP，数据分析，人工智能，网络爬虫等多种场合，功能十分强大。

JavaScript支持面向对象程序设计，指令式编程，以及函数式编程，它目前已经被世界上的绝大多数网站所使用，也被世界主流浏览器（Chrome、IE、Firefox、Safari、Opera）所支持[4]。

最初JS主要被用作浏览器的脚本控制语言，不需要服务器的支持，这样可以减轻对服务器的依赖，从而使服务器负担减轻。但是这也同时带来了严重的安全行问题，后来随着服务器性能的不断提高，程序员为了保证安全性，会优先选择将JS脚本保存在服务器上，在浏览器加载HTML文件时，会向服务器发送对JS脚本文件的请求，服务器会将JS文件返回给浏览器，从而切断了从外部改写JS脚本文件的途径，使得安全性得到了保障。再到后来，随着浏览器引擎技术，例如V8的发展，诞生了Node.js框架，由于其拥有事件驱动和异步IO等特性，JS也被用来编写服务器端的程序。

为了解决适配性问题，网景公司与欧洲计算机制造商协会推出了JS的新一代标准，ECMAScript6，简称为ES6，在这一版本之进行了大量优化与整合，使得JS变得更为现代化与人性化，添加了类和模块的语法，并且借鉴了Python风格的生成器和生成器表达式，箭头函数，promise和proxy功能，由于过多的“借鉴”，JS也广受批评，JS创始人调侃ES6为“成功的部分并不原创，原创的部分并不成功”。

在本次项目中会大量使用JS代码对网页效果以及图表状态进行控制，几乎所有的JS代码都为ES6标准，并且会引用许多JS第三方库，例如jQuery，Echarts，flexible.js等。

#### ​1.2.4 Echarts与图表绘制

Echarts是一个完全由JS编写成的用于数据可视化的图表组件库，它内置了许多美观，规范，高度可个性化定制的数据可视化图表[5]。Echarts最初由百度编写完成，并在2018年并入Apache基金会。Echarts的适应性很强，可以在大多数主流浏览器完美运行。

Echarts内置了各种常用于统计分析的图表，例如柱状图，折线图，散点图等等，还支持更多的复杂图表，如漏斗图，地图等等。

在使用Echarts包时，需要先下载Echarts源码或者使用经过CDN加速技术处理过后的链接在线引入，本项目采取提前下载源码本地引入的方法。另外一点需要注意的是，如果项目中使用了中国地图模块，则需要在引入Echarts之前，先引入China.js文件，这是因为为了简化Echarts文件的体积，一些数据文件被分割出去独立成一个文件，这其中就包括世界上各个国家的地图信息。

Echarts图表的样式与属性是一个由一系列键值对组成的JS对象所控制的，每一个键值对会对图表的某一个属性进行控制，如x轴，y轴，标题，颜色等等。在按照规则将这个JS对象编写完成后，需要将这个对象与图表实体相关联。

值得注意的一点是，在将图表与HTML中的盒子关联时，HTML中的那个盒子必须拥有确定的宽高，这个宽高可以是CSS中指定的宽高，也可以是flex布局自动推断的宽高，并且由JS类选择器选择到的盒子只能是单一的一个盒子，而不能是一组盒子。

### 1.3 基于Python的Django框架概述

#### ​1.3.1 Django技术简介

Django是一个基于Python语言开发的包，是一个专门用于快速开发可用的网站的WEB框架，它具有实用、简介、安全的特性。Django的出现大大简化了开发人员开发一个网站的流程，使得开发人员可以将更多精力放在新的功能的开发上，而不是在已经存在的功能上[6]。

本项目的后端部分完全由Django构造，使用Django有诸多好处，对于开发者来说，最突出的一个优点便是Django拥有十分庞大的社区。得益于Python语言的火热，基于Python的Django框架也受到了很多人的欢迎，在网络上有丰富的关于Django的文档，在著名的编程交流平台StackOverflow上，关于Django的问答也很多，这意味着在使用Django中出现的问题大概率可以得到有效的解决。并且，由于Python良好的可移植性，Django也可以在多种操作系统中正常运行，这使我可以在Windows下编写和测试，最后将代码移植到Linux服务器中。

#### ​1.3.2 Django项目结构

Django把一个网站的所有内容称作一个project，每一个模块称作app，这里的一个模块可以是单独的一个网页，也可以是一系列相关联的网页。

每一个Django项目的根目录都会有一个manage.py文件，这个文件是用来管理该项目的文件，也是Django程序的入口，在完成全部代码的编写之后，将会从这个文件开始执行，从而创建出一个网站。除此之外，manage.py还会将项目的包放在sys.path中，它将DJANGO\_SETTINGS\_MODULE环境变量设置为了setting.py文件的路径。

在刚刚创建好的Django中就已经存在一个app，它的名字与项目的名字一致，它代表了一个默认网页，也就是在初次启动Django后，通过IP:PORT访问这个项目时会展示的一个默认界面，界面的内容是Django默认的一个页面，我们一般不使用这个界面。在此文件夹下，有一个名为urls.py的文件，可以在此处修改默认路由，一般会把默认路由设置为导向我们后来新创建的一个app。在此文件夹之下，还有一个重要的文件，名叫settings.py，关于网站的一些重要的配置都在这里进行，例如主机过滤、app注册、模板配置、数据库配置、wsgi配置、静态资源位置等等。

#### ​1.3.3 Django的使用与配置

在使用Django框架之前，应该先安装好python，并且任意合适的编辑器，我选择的是vscode，全都准备好并且确定python所在位置在环境变量的PATH中之后，就可以打开命令行（可以是PowerShell，也可以是cmd），输入命令“pip install django”下载Django。下载完成后输入“django-admin startproject xxxx”，其中xxxx是项目名称。

创建好项目之后，进入项目文件夹，在文件夹内打开命令行，运行“python manage.py startapp xxapp”，其中xxapp是app的名字，运行完毕之后，会出现一个名为xxapp的文件夹，在编写这个app之前，需要打开xxxx文件夹之下的urls.py文件，在urlpatterns这个列表里写入“path(‘’, include(‘xxapp.urls’))”这一项，并且在文件最开头导入include模块，这个模块在django.urls内。

更改完urls之后，还需要将新建app进行注册操作，需要进入settings.py文件，找到INSTALLED\_APPS列表，将新建的app名字追加进去。这样就初步完成了一个新app的创建、路由更改与注册。

完成上述工作之后，项目还不能正常工作，还需要进入新建的app的文件夹，在app目录下的urls.py内，找到urlptterns列表，写入“path(‘para’,view.func,name=’func’)”，这意味着如果用户访问了IP:PORT\para 路径，将会转到view.py文件下的func方法进行处理。这时进入view.py文件创建名为func的函数，该函数接收一个参数，一般命名为request，它是浏览器在请求时发送的http请求参数，在这个函数体内部，可以对要返回的数据进行一定的处理或者是逻辑控制，最后通过return 返回经过render方法处理后的HTML文件（render方法需要在文件头部引入，它在django.shortcuts内部）。这样，在浏览器上输入正确的网址之后，就会看到浏览器将返回的HTML页面渲染成的网页了。

### 本文主要内容

本项目通过使用Web前端技术结合后端Django技术实现一个数据可视化的网站展示平台，并且最终将项目部署到阿里云服务器上，实现在用户计算机上的浏览器内输入相应的IP地址，就可以看到精致的网页。

本文将通过前端技术、后端技术、服务器配置的顺序，分模块逐步制作完成一个完整的网站，在这个过程中，会依次介绍多种目前业界流行的技术，诸如jQuery、Ajax、echarts等等，接着会着重介绍目前比较火热的云服务器技术，并且详细的讲解了在一台服务器上如何使用Nginx和uWSGI两个服务器软件将一个Django项目完美的运行起来；在文章的结尾又给出了本次项目的缺点，以及在未来如果出现更复杂的需求该如何重构本项目的代码以满足需求。

## 2 前端页面的结构搭建与美化

### 2.1 主界面的设计与布局

#### 2.1.1 主界面头部制作

主界面的头部主要用于承载整个页面的标题，并且将会在右上角制作一个当前时间功能。首先创建一个index.html文件，再vscode下在文件头部输入一个英文的“!”再按下回车就会自动生成页面的准备信息。按照HTML5规范，头部应该用一个header标签包裹起来，作为一个整体，放在body标签之内，在header标签内部，首先是一个h1标签，用于显示整个标题，紧接着就是一个div盒子，用来放显示时间的文字，为了后期在JS脚本里控制这个div，需要给div添加一个类，这里取名为“showTime”。

为了与后面的页面主要部分相区分，在这里先提前开始编写这个模块的CSS和JS内容。另外新建一个index.css文件，由于HTML标签在初始时都具有一些默认样式，例如h1标签的文本会被加大加黑等等，为了方便后期统一设定，需要在index.css文件首部将这些样式统统取消。取消完成之后，给body添加一个背景颜色#333，并且给header标签添加背景图片，给header内的h1标签设置字号，颜色等等，并且让文本剧中。对于showTime模块，将其强制定位至右上角，并且缩小字号，设置颜色，控制上右距离。

最后，需要用JS脚本控制时间的数值自动更新。创建一个getTime.js，并且在showTime模块下面用script标签引入getTime.js文件。在getTime文件内部写入一个等间隔任务计时器“setInterval”函数，这个函数接收两个参数，第二个参数是一个整数，代表每两次任务之间的间隔，单位为毫秒，在这里应该设置为1000，即为1秒，第一个参数是一个函数（回调函数），代表每隔1秒将会执行一次这个函数，根据JS语法，这个函数可以写成匿名函数或者箭头函数。在这个函数里面，新建一个Date类，并且获取年、月、日，将它们转化为字符串并且拼接成一个用户友好的句子。用“document.querySelector(“.showTime”)”类选择器函数获取HTML元素，并将它的innerHTML属性设置为制作好的字符串。

#### 2.1.2 主界面主体布局

目前的页面主流布局方式有三种：传统布局，Flex布局，Grid布局。传统布局对每一个盒子精确的像素级别的位置控制，能达到最大的准确性，但是也最复杂；Grid布局将页面看成一个一个的网格，对每一个网格进行控制，开发难度降低但是对于一些不规则的结构很难应用；Flex布局介于二者之间，本次布局就使用了Flex布局[7]。

布局思路为：先将整个页面看作一个整体，在这个整体内部按照左中右三列按比例分配空间，然后再将每一列看作一个整体，在这个整体内部对空间按需求分配。

在body内部新建section标签，类名为mainbox，作为整个主题部分，在mainbox内部新建三个div类名都为column，在第一个和第三个column内部都新建三个div，类名为panel，在每一个panel内部都新建一个h2用于写表格的标题和一个div类名为chart用于放表格还有一个div类名叫panel-footer用于制作特效。

在index.css文件内，将mainbox设置为弹性盒，即添加flex属性，这样，它内部的盒子都会收到flex布局的控制。mainbox内部的三个column的宽度比例为3：5：3，用选择器选到第一个和第三个column，为它们添加flex：3，为第二个column添加flex：5。第一第三个column内部的三个盒子会沿着高度方向均匀分布，这时再给每个panel盒子设置一个虚化的背景和边距，就可以看到每个panel的轮廓了。

对于第二个column，可以将其内部分为两部分上部分为数据展示模块，类名记为num，下部分为中国地图模块，类名为map。在num里面创建两个列表项分别显示供应电量和需求电量，这两个数字将来会从数据库内部读取。

### 2.2 页面美化

#### 2.2.1 用Less替代CSS

CSS文件专门用于对HTML文件进行样式设定，在编写css代码的时候需要遵守一定的规则，即每个HTML元素的属性都需要并列书写，并且对于多级嵌套的元素也是如此，但是这并不符合HTML文件的嵌套式结构，增加了编程的困难，在这个背景之下，Less诞生了。Less完全兼容CSS的语法，并且支持了嵌套式编写代码，使程序员不必再在HTML元素的父子关系上花太多的精力，在使用Less之前需要在vscode里下载less预处理插件，下载完毕之后，可以在css文件夹内创建.less文件，在编写完less文件并保存之后，会自动生成一个同名的css文件，也就是说，预处理器自动按照规则将less生成了等价的css文件，在HTML文件里依然引入css文件。

#### 2.2.2 CSS属性设置

在CSS文件内部可以对HTML元素的各种属性进行设置，但是在此之前，需要准确的定位到HTML中的某个元素，可以用元素的名字（例如div、section、header），元素的类名（如上文中提到的mainbox，column）和元素的id，以上方式分别是元素选择器、类选择器、id选择器。在选择到元素之后，就可以对元素的各种属性进行调整，可以设置盒子的大小、边框、内边距、外边距、背景图片、颜色等等，这里又许多的细节，但是限于篇幅原因，就仅作简单介绍。

#### 2.2.3 盒子的特殊四角

现在要将主界面的每一个盒子（除了地图盒子）的四个角都设置为加粗并且有其他颜色的边框，但是这种效果只依靠CSS调整属性是无法实现的，这就引出了伪元素的概念。

在HTML的盒子模型规则中，在最开始每一个盒子元素内部都有两个“隐形”的盒子，它们内容为空，且大小为零，它们分别是div::before和div::after。将这两个盒子设置一个比较小的长和宽，设置它们的背景颜色为透明，并且设置实线有颜色的边框，再通过绝对定位的方式将他们定位到大盒子的左上角和右上角，最后对于左上角的小盒子，将它的右边框和下边框设置为透明；对于右上角的小盒子，将它的左边框和下边框设置为透明，这样就实现了左上角和右上角的特殊边框。

在本项目中，可以将上文提到的panel盒子内部的伪元素设置为左上和右上的边框，对于左下角和右下角的边框就用刚才同时添加的panel-footer盒子，设置左下角和右下角的边框的步骤于左上与右上的过程类似。这样，界面上的盒子就获得了四个角的边框，使得每个盒子之间的界限变得更加明显，大大增加了美观性。



图2-1

### 2.3 图表引入界面

#### 2.3.1 Echarts引入与配置

在官网上下载完echarts.js之后就可以用script标签将它引入了，但是一定要在index.js之前引入，引入完成之后，用echarts.init()函数创建一个图表对象，这个函数需要一个HTML元素作为参数，可以用上文提到过的document.querySelector()获取，这个函数支持一切CSS选择器的规则。此函数执行完毕之后便创建了一个图表对象，但是此时这个图表没有任何配置，需要新建一个JS对象，利用键值对的方式为这个图表进行各类配置，各种配置的规则可以根据官方文档进行学习。

在将配置项编写完成之后，就可调用图表对象的setOption()方法，将配置对象传进去，之后就会在页面的盒子上显示出图标了。为了在网页大小改变时图标的大小也能自适应，可以添加一个网页窗口大小监听器，即代码“window.addEventListener(“resize”,function)”在第二个函数参数内部写入chart.resize()，这样就完成了图标的自适应功能。

在图表的配置项中，有几个是值得注意的。tooltip选项能够控制当鼠标放在图表上时图表的行为，例如对于条形图来说，可以将其设置为当鼠标放在某一个条时，该条出现选择特效，并且出现侧边框，显示该项的相关信息；grid选项能够控制在一个盒子内部，图表实际大小以及图表所在的位置；series是一个由多个对象组成的数组，数组中的每一个对象都代表了图表上的某一部分，例如对于条形图来说，每一个对象就代表一组方块，对于折线图来说，它就代表一条折线，对它进行配置可以设置数据的展示形式，要展示的数据本身也要放在series的data项目中。

#### 2.3.2 Ajax技术与局部刷新

在WEB技术刚刚出现的早期，每当一个网页想要展示新的内容的时候，只能是通过超链接跳转到一个新网页或者是将本网页全部更新，但是很多时候只是需要更新很少的信息，例如某些论坛的评论区，但是技术限制使得只能全部更新，这造成了网络资源的浪费并且使得用户的使用体验下降，直到Ajax技术的诞生。

Ajax技术（中文译为异步的JavaScript和XML）可以快速将更新的部分显示在网页上而不需要重新加载整个页面，这让网页可以更快的回应用户的交互[8]。在本次项目中，所有图表所要展示的数据都是用Ajax技术从后台获取然后再渲染在浏览器上的，在创建所有图表之前，需要用ajax向后台发请求，发请求的方式有两种，第一种是用JavaScript自带的方式，但是这种方式需要自主配置许多配置项，比较麻烦；另一种流行的方法是用一个JS集成库jQuery中的ajax函数，在使用jQuery之前，需要去官网下载jQuery代码包并且引入jQuery，jQuery中有一个主对象jquery，简称为“$”,可以通过“$”加点调用其内部的函数。

1. $.ajax({
2. method: "GET",
3. url: "/api/data",
4. async: false,
5. data: "35",
6. success: deploy,
7. error: function (error) {
8. console.log(error);
9. },
10. });

调用$.ajax()需要传入一个对象作为参数，其中method是http请求的请求方式，“GET”表示从服务器拿数据；url代表数据在网站的存放路径（详见第三章）；async选项表示是否异步，如果不设置默认为true，如果为true，代表正常代码执行流不会等待该函数请求数据，会直接跳过此函数；data代表请求数据时要传递的参数，35代表全国数据（详见第三章）；success和error代表请求成功和失败的时候要执行的函数，成功时要执行deploy函数，该函数将请求到的数据配置给各个配置项。在这之后，当代码执行到setOption()时，就将会渲染所有的图表。

### 2.4 中国地图页面引入界面

这个模块是整个页面里最复杂的部分，但是幸运的是，地图的绘制工作已经集成完毕了，我们只需要引入并且进行配置就可以使用了。地图模块是基于SVG技术绘制而成的，在使用之前，需要去官网上下载china.js文件，并且在index.js文件之前引入。引入完毕之后，在index文件内用echarts.init()创建一个图表对象，并且将option配置项里的series[0]中的type设置为map，将maptype设置为china，这样一张中国地图就会在屏幕中央的盒子里出现了。

为了更加契合主题，需要对地图的样式进行设置。首先设置鼠标交互效果，将tooltip设置触发方式为item，随后再在legend选项里添加selectedMode为single，这保证了当鼠标点击地图上的某个省份时，只会有唯一一个省份的区域会高亮，在series[0]中的itemStyle选项里可以配置地图在某个条件下的颜色，在它里面的normal选项中的areaColor设置为初始颜色，即是地图刚刚出现时的颜色，在与normal并列的emphasis选项中的areaColor中可以设置当鼠标点中一个区域时它的颜色，这里设置为一个比normal更深的颜色。

最后，设置zoom选项为1.2，即将地图放大1.2倍，再将roam设置为true，设置为true可以让地图被鼠标拖动，也可以通过鼠标滚轮放大或者缩小。

### 2.5 地图点击功能实现

如果要实现鼠标选中地图上的某一个省，地图两边的图表所展示的数据就变为这个选中的省的数据如图2-2所示，首先需要监听地图点击事件的发生。Echarts内置了事件监听函数，通过chart.on(“click”,func)就能实现点击事件的监听，func是点击事件发生时的要执行的函数，这个函数需要接收一个参数param，它由浏览器返回，可以在函数体内部直接使用。



图2-2

param是一个JS对象，在其内部有一个名为dataIndex的属性，根据每个省的不同，其值也不同，例如如果点击了新疆省，返回的param的dataIndex就为27。dataIndex的范围是0~34，在这里，我们默认的将什么都没选的默认状态设定为35，也就是全中国的dataIndex为35，这样，在第一次加载界面时的ajax请求就可以传递35这个参数，时的后端查询数据库的操作变得统一。

在这里的地图点击事件就会出现以下几种情况：1、最初加载时未点击任何省份；2、点击了某一个省份；3、点击了一个已经选中的省份；4、点击了另外一个省份。对于这四种情况，可以统一用以下这段代码实现。

1. let selected\_id = 35;
2. let last\_id = 35;
3. myChart\_map.on("click", function (params) {
4. selected\_id = params.dataIndex;
5. selected\_id == last\_id
6. ? (selected\_id = last\_id = 35)
7. : (last\_id = selected\_id);
8. console.log(selected\_id);
9. // 后续逻辑 …
10. });

在监听函数执行之前，先设置了selected\_id和last\_id，分别代表了目前选择的省份id和上一个选中的省份id，它们的初始值都是35，这意味着当前和上一个选中的省份都是全国。在进入监听函数之后，令selected\_id为选中省份的dataIndex，然后进行判断，如果当前选中的id与上一个id相等，则说明遇到了情况3，则将所有数据恢复到全国状态，即是发送序号35的请求，如果不是，则是情况2或者情况4，则将last\_id置为当前id以供下一次点击事件发生时使用。

随后便可以通过ajax和selected\_id向后台发送请求了，将相应的数据取到之后，需要将它们分配给每一个配置项，分配完毕之后，对每一个图表都再执行一次chart.setOption(option)，就可以看到图表再页面上更新了。

### 2.6 二级页面制作

#### 2.6.1 进入二级页面

为了更详细的展示数据，需要增加一个图表的详细页面，这个页面会通过点击页面上的六个图表中的任何一个图表进入，实现这个功能需要为每一个图表添加点击事件。新建一个click.js文件并在HTML文件内通过script引入，在这个文件内通过选择器获取每一个图表的HTML元素，然后对每一个元素都添加一个click事件，当click事件发生之后，通过window.open()函数打开一个新的页面，调用时需要将要打开的HTML文件名传入。

#### 2.6.2 页面设置

对于这个页面的布局与美化，可以用到与主页面差不多的流程，在布局方面同样使用flex布局，只不过在这个页面中，将页面均匀分为三个相等的列，然后每个列又都均匀分为三个均等的盒子，对于header部分，直接复用主页面的header以保障一致性即可。这样整个页面就会有九个图表，再通过选择器对每一个盒子设置不同的图表。制作完成后的页面如图2-3所示。

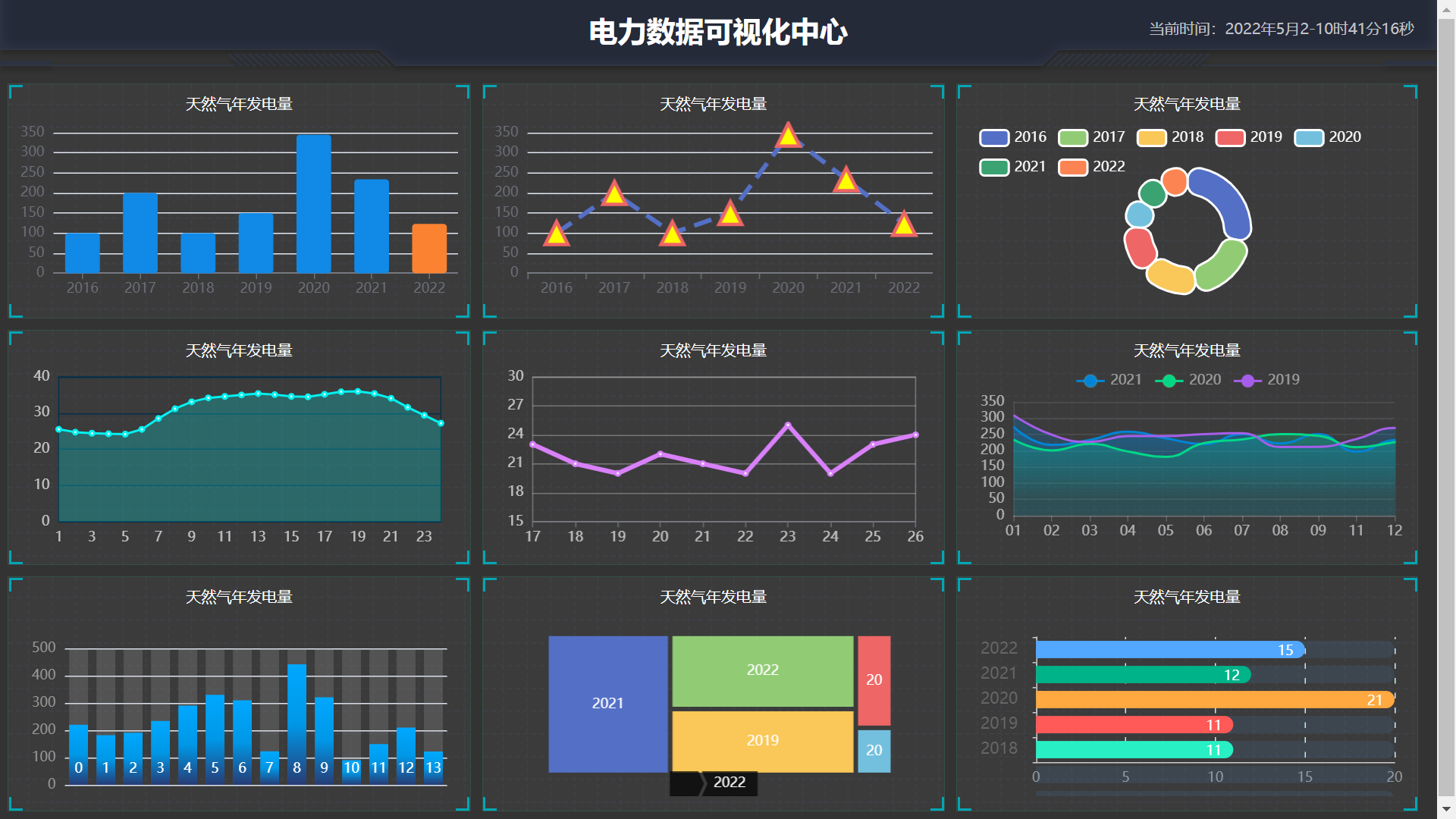


图2-3

## 后端配置以及设置路由

### 3.1 启动项目

在下载完Django后就可以创建项目并且用vscode打开，通过python manage.py startapp datacenter命令就可以新建一个app。运行完毕，打开命令行，输入python manage.py runserver 启动项目，在浏览器里访问127.0.0.1:8080就可以看到Django项目的初始页面。

### 3.2 路由配置

在测试完项目可用之后，就可以进行路由配置了，前端共有两个页面，需要为它们分别配置路由。首先在mysite文件夹下的urls里的urlpatterns里用path函数引入datacenter的urls文件，路径为“”，即为空，代表默认时就转到datacenter的urls再进行查询。

设置完成之后，进入datacenter文件夹，在urls文件里的urlpatterns中添加path(‘’,views.index,name=“index”)和path(‘z1’,views.z1,name=“z1”)，添加完成之后，进入views文件，创建index函数和z1函数，在这两个函数内部返回通过render函数处理的相应的HTML文件，即 return render(request, ‘index.html’,)。

另外，值得注意的是，这两个文件必须放在datacenter文件夹内的templates文件夹下，如果不是在这个位置，Django将无法找到文件。

### 3.3 静态资源定位

在前端模块里，除了HTML文件，其他的文件都被称为静态资源。如果不进行设置，只在配置路由之后打开网页，只能看到HTML结构而看不到CSS美化效果和JS的交互效果。解决方法是在整个项目的根目录下创建一个static文件夹，将所有的JS文件、CSS文件、图片文件、字体文件都放进去，然后打开mysite文件夹下的setting文件，新建一个名为STATIC\_URL的变量将其赋值为“static/”，这就为程序指明了静态资源存放的路径，当程序检索HTML文件时，遇到请求静态资源的代码就会到根目录下的static文件夹下寻找相应文件，从而完成渲染页面。

### 3.4 数据库设置

由于后端所处理的所有数据都将是有数据库所管理的，所以要为Django项目配置数据库。Django默认支持的数据库sqllite3，但是本次项目决定使用MySQL数据库，所以需要更换数据库。首先应该安装好MySQL数据库并且打开它（详见4.2），然后进入mysite文件夹，打开settings文件，在其内部找到DATABASE变量对应的对象，将其ENGINE项改为‘django.db.backends.mysql’，将NAME项改为数据库的名字geodata，HOST设为127.0.0.1，Port设置为3306，USER和PASSWORD设置为新建数据库是创建的用户名和密码。

在做完上述工作之后，需要在命令行运行 python manage.py makemigrations命令生成迁移文件，然后运行python manage.py migrate 进行迁移。在这之后数据库就和django程序关联了。

### 3.5 数据接口编写

本次项目中的前端图表所展示的数据都是要从后端获取的，要完成这项操作，可以为网站增加一个获取数据的路由，每当前端访问这个路径的时候就将数据打包为JSON格式返回。首先在datacenter文件夹的urlpatterns里增加path(“api/data”, views.get\_data, name=“api-data”)条目，这代表当访问了127.0.0.1:8080/api/data时会获取到views文件里get\_data函数所返回的数据。

在views文件里，新建一个get\_data函数，该函数接收一个参数request，该参数内部包含了浏览器发送的请求信息，其中一个重要的参数就是所点击的地图省份的geoIndex，获取到此id就可以根据它从数据库中查询到相应的数据。

在查询数据库之前，需要从django.db中引入connection模块，然后通过SQL语句从数据中读取数据。

1. def get\_data(request):
2. import json
3. params = list(request.GET.keys())
4. geoIndex = int(params[0])
5. mycursor = connection.cursor()
6. mycursor.execute("select \* from data")
7. rows = mycursor.fetchall()
8. rows = dict(zip(range(36), (k[1] for k in rows)))
9. dataX = json.loads(rows[geoIndex])
10. return JsonResponse(dataX)

从数据库里读出的数据是以字符串的形式存储的，并且格式不符合要求，需要对其格式进行处理，并且通过json模块将其转化为json格式，最后根据id返回前端所需要的数据段。在完成上述所有的设置之后，整个项目的功能流程就变得清晰起来了，如图3-1所示。



图3-1

## 4 服务器配置

### 4.1 云服务器配置

#### ​4.1.1 阿里云服务器简介

云服务器，英文简称为ECS，从使用的角度来说，就是一个服务器主机，所谓的服务器也就是一台电脑，只不过云服务器所使用的电脑的配置和结构要比我们平常使用的电脑优秀的多[9]。云服务器的本质是对硬件设备利用软件技术进行的虚拟，虚拟出来的服务器在运行的时候所占用的也是实体服务器的资源。

阿里云在国内的云计算行业内处于领先地位，故本次项目会使用阿里云服务器进行配置和部署。在阿里云服务器里有许多抽象的概念，例如实例、部署集、快照、安全组等等，其中在本次项目中的一个比较重要的概念就是安全组。

在windows系统里有为了保护电脑不被侵害的系统防火墙，云服务器中的安全组的作用与其类似，它规定了只用一些设定的IP和端口号可以访问云服务器。在购买完云服务器之后就可以进入配置界面进行配置，在本次项目中需要进行5个配置，分别是0.0.0.0/22，打开这个端口用于开放SSH连接服务（详见4.1.3）；0.0.0.0/80，打开这个端口用于开放http服务；0.0.0.0/3389，这个端口被默认打开，用于实现远程显示协议（RDP）；0.0.0.0/8000，用于开放django服务；0.0.0.0/3306，用于开放MySQL数据库服务。

#### ​4.1.2 Linux的基本配置

从理论上来说，云服务器上可以运行任何一种操作系统，但是考虑到云服务器的用途，大多数情况下还是会选择基于Linux内核的操作系统。

Linux内核代码早已经被开源，现如今，以Linux内核为基础开发的操作系统的种类越来越多，比较出名的有RedHat、CentOS、Ubuntu、Arch、deepin、kali等等，甚至每个人都可以按照个人喜好开发自己的操作系统，本次项目中选择的是Ubuntu20.04LTS版本的操作系统。

在云服务器管理界面可以选择创建实例，创建完成之后就会出现操作系统的选择界面，点击选择Ubuntu就可以配置成功了。

在云服务器上的Ubuntu系统默认使用命令行进行控制，这个程序被称为shell，在命令行内输入的命令称为shell命令，由命令组成的文件称为shell脚本。在刚刚进入操作系统时，命令行会提示输入账号密码进行登录，登录后会使用默认用户root，root的权限非常大，可以执行操作系统里的所有命令，容易造成安全隐患，如果是日常使用Linux则不建议用root用户，但是作者对Linux操作系统足够熟悉，所以本次就直接使用root用户对系统进行配置。

为了能让该系统运行django程序，我们需要提前下载相关的软件包，但是由于特殊原因，在国内用Linux下载软件的速度特别慢甚至有可能失败，所以要先进行换源操作，也就是更换软件的提供者。首先运行 vi /etc/apt/sources.list，运行成功后会进入vim编辑器界面，vim编辑器的操作与大多数编辑器不同，进入编辑界面后默认为“命令模式”，点击键盘上的i键就会进入“插入模式”，通过移动键和退格键将所有内容删除，然后将以下内容复制到文件内部。

1. deb http://mirrors.cloud.aliyuncs.com/ubuntu/ focal main restricted
2. deb-src http://mirrors.cloud.aliyuncs.com/ubuntu/ focal main restricted
3. deb http://mirrors.cloud.aliyuncs.com/ubuntu/ focal-updates main restricted
4. deb-src http://mirrors.cloud.aliyuncs.com/ubuntu/ focal-updates main restricted
5. deb http://mirrors.cloud.aliyuncs.com/ubuntu/ focal universe
6. deb-src http://mirrors.cloud.aliyuncs.com/ubuntu/ focal universe

复制完毕之后，点击ESC键，然后点击“shift+:”，输入wq后点击回车，这样就将文件内容成功修改并且保存，换源工作就完成了。

紧接着在命令行上输入apt-get update 命令更新软件列表，然后输入apt-get upgrade更新软件包。自此，软件安装的准备工作就完成了。运行apt-get install python3 安装最新的python，安装完成后就可以使用pip命令了。运行pip install django安装django，运行pip install mysqlclient安装django中MySQL管理程序。

#### ​4.1.3 Xshell远程连接云服务器

在上一节中的所有命令行操作都是在浏览器上完成的，如果网速并不是特别快，就会感受到明显的卡顿，所以在浏览器运行shell不是长久之计，所以接下来的工作会在windows端的Xshell软件内完成。

在官网上可以下载到社区免费版本的Xshell，安装完成并且打开后会提示进行配置，在这里跳过配置，因为一会儿会在命令行上进行配置。在出现的命令行上输入 ssh 8.130.104.218@root命令，这代表要用ssh协议以root用户登录该IP所代表的主机，这时命令行会提示输入密码，输入完毕之后会弹出一个对话框，该对话框显示了一个ssh密钥，点击保存按钮将该密钥保存，使该设备以后登录这个IP时不需要输入密码。完成后就已经连接到该主机了，可以运行pwd命令查看自己的位置，如果成功显示一个路径，则说明可以正常使用了。

### 4.2 MySQL数据库搭建与配置

#### ​4.2.1 MySQL简介

MySQL是一个关系型数据库管理系统，它被广泛的应用在WEB开发领域，因为MySQL具有结构简单、响应速度快、开发成本低等特点，所以其被众多的中小型和大型网站所使用[10]。由于MySQL使用C语言和C++编写，所以它可以完美的支持Linux系统，并且MySQL使用标准的SQL语句进行增删查改的操作，这意味着如果用户在之前使用过其他类似的数据库，例如Oracle、DB2、SQL server等等，基本上可以无障碍上手。

#### ​4.2.2 MySQL的下载与配置

在Ubuntu系统下的命令行里直接运行 apt install mysql-server 就可以直接下载MySQL的最新版本，并且环境变量（例如PATH）也将自动配置好。下载完毕之后，运行mysql –version 如果安装成功就会显示出mysql的版本信息。

运行mysql -u root -p登录数据库，目前数据库的root用户是没有密码的，当提示输入密码时直接按回车就可以登录进入数据库，进入后输入set password for root@localhost = password(‘新密码’)。这样就为新数据库设置了一个root用户。最后，不要忘记在Django项目的mysite文件夹下的settings文件夹的DATABASE配置项中添加数据库的账号、密码、名字以及IP地址。

#### ​4.2.3 使用Navicat可视化工具对数据库进行管理

MySQL完全可以用命令行进行控制和使用，但是这并不方便，为了更快速的开发，可以选择Navicat工具远程操控MySQL。Navicat是一套可以同时发起多个连接的数据库管理工具，它的功能很强大，强大到可以满足专业人员的一切需求，同时又对数据库服务器初学者相当友好。

在官网上可以下载到Navicat的30天免费使用版本，下载完成后根据提示进行安装，安装完成后打开进入主界面，此时主界面为空。点击左上角的“连接”图标可以创建一个新的连接，此时提示输入连接名，这里可以随便起一个名字，主机选项就填写云服务器的公网IP，端口使用默认的3306端口，该端口已经在云服务器的安全组里开放，用户名和密码就填写在之前注册的root和其密码。

连接完成之后右键点击连接选择新建数据库，起名为data，然后再右键点击data新建表，起名为geodata，建立完成后为表添加索引，索引为0-35数字，数据类型为JSON，但是数据库会将JSON保存为字符串。然后就可以在可视化界面为表添加数据了。

### 4.3 服务器软件的配置

#### 4.3.1 uWSGI的配置与使用

uWSGI用于实现Django程序和服务器之间的交互，它是一个实现了WSGI协议的WEB服务器软件，WSGI协议是一个规定了WEB服务器应该如何与应用程序（Django、Flask）通信的规范的协议，uWSGI所在的层次如图4-1所示。所以uWSGI用于实现本次项目中的Django程序与云服务器之间的连接[11]。

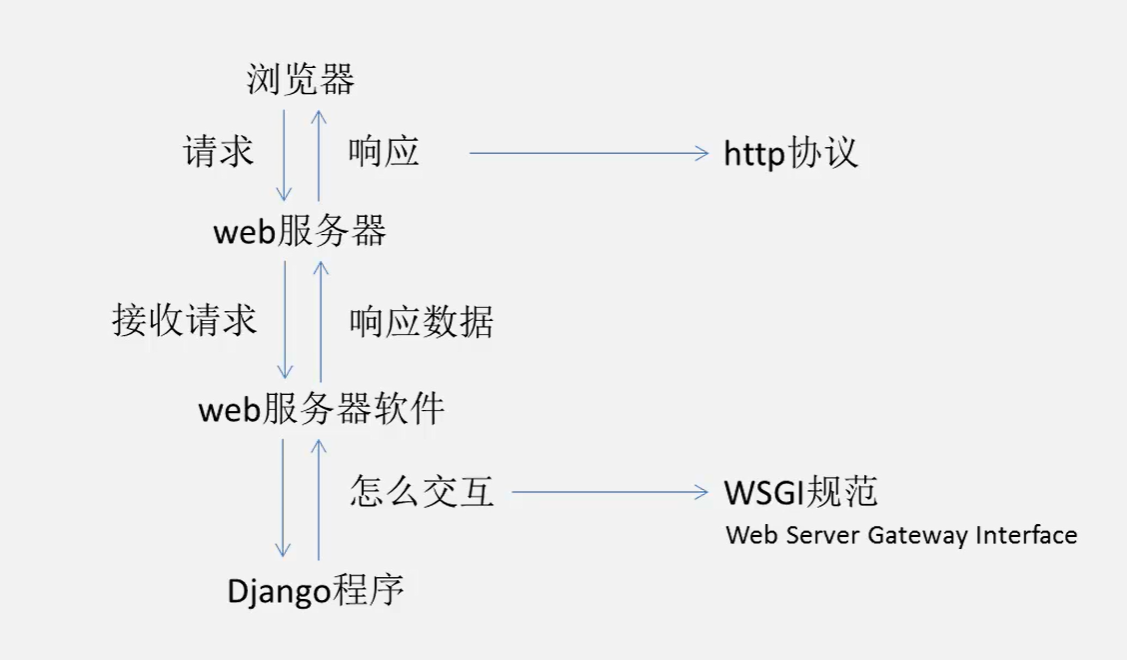


图4-1

在Xshell中输入pip install uwsgi 就可以安装此软件。为了检测安装效果，需要进行测试，使用cd~命令进入root目录内，创建一个test.py文件，在其内部写入以下内容：

1. def application (env, start response):
2. start\_response("200 ok", [("Content-Type", "text/html")])
3. return [b"Hello World"]

写入完成后保存退出，在命令行内运行 uwsgi –http :8000 –wsgi-file test.py ，这时命令行会出现很多提示信息，如果没有报错，就可以在本机的浏览器上访问IP+8000端口号，这时页面上会出现Hello World字样，这说明测试成功，uWSGI可以正常使用。 接下来就可以用uwsgi配置django程序了，在命令行内输入uwsgi --chdir /root/mysite --http :8000 --module mysite.wsgi:application，这样就完成了对项目的初步配置，此时访问服务器可以看到页面的结构，接下来进行的nginx服务器配置会完善整个界面。

#### ​4.3.2Nginx的配置与使用

Nginx英文全称为Engine X，是一个轻量级的高性能WEB服务器软件[12]。它可以在大多数的Linux系统上运行，在命令行里输入apt install nginx命令可以直接下载nginx，下载完成后就开始对nginx进行配置。

首先使用cd /etc/nginx/命令进入nginx所在的目录，其中有两个文件夹是需要配置的，一个是sites-available，这是nginx的一些可用的配置，另一个是sites-enabled，这是nginx的一些已经启用的配置。先用命令cd sites-available进入该文件夹，在该文件下，用vim mysite.conf命令新建一个配置文件并且进入编辑状态，输入以下内容：

1. server {
2. listen 80;
3. server\_name mysite;
4. charset utf-8;
5. client\_max\_body\_size 75M;
6. location /static {
7. alias /root/mysite/static;
8. }
9. location / {
10. uwsgi\_pass 127.0.0.1:8000;
11. include /etc/nginx/uwsgi\_params;
12. }
13. }

输入完毕后保存退出，然后输入cd /root命令回到家目录，再输入mkdir mysite\_uwsgi命令创建一个配置文件夹，进入该目录，输入 vim mysite.ini创建一个配置文件，在该文件内输入以下内容：

1. [uwsgi]
2. chdir = /root/mysite
3. module = mysite.wsgi:application
4. master = True
5. process = 4
6. harakiri = 60
7. max-request = 5000
8. socket = 127.0.0.1:8000
9. uid = 1000
10. gid = 2000
11. pidfile = /root/mysite\_uwsgi/master.pid
12. daemonize = /root/mysite\_uwsgi/mysite.log
13. vacuum = True

输入完成后保存退出，在命令行上输入 uwsgi --ini /root/mysite\_uwsgi/mysite.ini 命令，此时服务已经启动了，可以通过命令 ps -aux | grep uwsgi查看进程，如果屏幕上显示有5个进程则说明启动成功。

启动uwsgi服务之后，再一次进入/etc/nginx/文件夹，输入ln -s /etc/nginx/sites-available/mysite.conf /etc/nginx/sites-enabled/mysite.conf，此命令会在sites-enabled下创建一个sites-available下mysite.conf文件的软链接，然后就可以重启nginx服务了，在重启之前，输入nginx -t命令查看nginx进程是否正常，如果显示syntax is ok，则说明可以正常使用了，然后就输入service nginx restart命令重启nginx服务，此时在本地浏览器上直接输入8.130.104.218，按下回车就可以看到精美的网站内容了。

## 5 设计方案的影响分析

本课题通过对制作一个对电力数据进行可视化展示的web应用，加深了对数据可视化的理解，了解了电力数据背后所存在的对于社会诸多方面的反映，包括环保、健康、经济等等。

每个国家的电力统计数据都一定程度上表现了了这个国家的环境情况，例如各个发电途径来源的占比组成可以反映出该地区的能源结构是否优秀，如果其中核能，风能太阳能等先进的能源占比较高，化石燃料发电的占比较低，则说明该地对于能源的利用相比其他地区更加先进，对环境的污染更小；将过去各年的碳排放量和2030年的碳排放量目标放在一起可以直观的看出每年需要减少的碳排放量，从而帮助环境管理部门制定下一年的决策。

对发电途径占比中的燃烧煤炭，石油等占比进行分析，可以判断出该地的有害气体排放对当地居民的健康状态的影响。上述几发电方式所产生的硫氧化物是大气的主要污染物之一，虽然发电厂在排放这些气体时会对它们进行处理，但是处理的效率并没有达到百分之百，它们被人体吸入后会导致人类的肺部疾病，对人的身体有很强的危害性，除此之外，它们还会对植物和动物造成危害，当它们在空气中变为酸雨时，也会对古建筑等历史古迹造成重大的影响，所以通过检测这几项指标，可以帮助人们提前预防灾害，也可以用作发电厂的参考指标。

对过去多年内某项资源发电的占比可以分析出世界经济的情况，例如分析每一年的石油发电量发现石油占比呈下降趋势，或者某几年该国家的石油发电量占比明显下降，则可以判断出该国家可能受到了国际政治经济的影响。再比如说可以根据过去多年的发电时天然气的使用量判断出接下来几年的天然气行情，可以帮助管理人员提前做出准备，如提前加购或者是减少购买量。

## 结论

本次项目全面应用了Web技术，通过Echarts框架结合众多的前端和后端技术实现了一个电力数据可视化中心的开发，实现了用多种形式展示数据得要求，并且做到了方便，美观。

对比其他可以实现本课题的技术，例如python的绘图工具，本项目具有外形更美观，使用更方便，形式更丰富的优点。Web技术中的HTML/CSS技术是目前技术界里为数不多的可以实现对页面的像素级别的控制的技术和基于强大的引擎所带来的流畅的动态效果，这是python绘图包或是诸如Qt等界面开发技术所无法比拟的；在使用时，只要在本机的浏览器上输入网址即可以浏览到数据分析的界面，不需要过多的顾虑不同用户电脑系统的差异，更不需要提前下载什么安装包或者是可执行程序，极大的优化了用户的体验；开源的Echarts框架提供了众多的图表格式，可以帮助用户以更加丰富多彩的形式观察数据的变化，并且该框架还拥有丰富的社区，用户可以看到其他人制作的精美的图表，为自己带来灵感，随时优化自己的图表，这些都是其他方法不能实现的。

即便如此，本项目也有一些不足之处。例如代码架构耦合度偏高，本项目只在逻辑上实现了前后端分离的设计，但是由于服务器数量的限制，没有在代码上也实现前后端分离，因此从效果上来看，前端的代码是嵌入在后端的Django文件夹内的。

从长远上来看，本项目可以升任在一些访问量较低，数据量不太大的场合内使用，如果将来的需求升级，需要整体优化代码结构，将前后端分离，并且需要优化数据库结构，将不同的数据请求按照结构化存储。

## 致谢

谢谢你，泰罗。

## 参考文献

[1] HACKER S, 苏子栋. HTML技术简介 [J]. 电子与电脑, 1998, (03): 90.

[2] 陈上. 基于HTML5技术的WEB前端框架的设计及应用 [J]. 科技创新与应用, 2021, 11(33): 63-6.

[3] 徐亚. 基于HTML5和CSS3的网页前端设计研究 [J]. 电脑知识与技术, 2022, 18(07): 61-2+70.

[4] 宋雅飞. JavaScript特效在企业网站设计中的应用 [J]. 无线互联科技, 2022, 19(02): 91-2.

[5] 周洪斌, 陈立平, 刘连浩. 基于ECharts的数据可视化应用 [J]. 沙洲职业工学院学报, 2021, 24(01): 3-9.

[6] 邱红丽, 张舒雅. 基于Django框架的web项目开发研究 [J]. 科学技术创新, 2021, (27): 97-8.

[7] 梁艳玲. 基于HTML+CSS网页布局定位参数研究 [J]. 吕梁学院学报, 2022, 12(02): 17-22.

[8] 王艳梅. jQuery在AJAX技术框架中页面加载的应用分析 [J]. 电脑知识与技术, 2020, 16(36): 210-2.

[9] 吕奎. 基于Django框架的发酵技术云服务器开发 [D]; 江南大学, 2021.

[10] 盛莉. 基于MySQL的数据管理平台设计 [J]. 信息与电脑(理论版), 2020, 32(22): 152-3.

[11] 王曦. 基于Django的微信营销系统的设计与实现 [D]; 东南大学, 2017.

[12] 王磊, 田宗梅. 基于Nginx的医院互联网应用网络架构实践 [J]. 中国数字医学, 2020, 15(03): 84-6.