**湖南工程职业技术学院**

**毕业设计**

**题 目** 胡润富豪榜数据采集、分析与可视化

**毕业设计类型** ☑产品设计 □工艺设计 □方案设计

**专 业 名 称**  大数据技术

**所 属 班 级**  2021031501

**学 生 姓 名**  何子骏

**学 生 学 号**  202103150125

**指 导 老 师 （学校）**  周杨淼

**指 导 老 师 （企业）**

目 录

[1项目概述 2](#_Toc1218)

[1.1背景介绍 2](#_Toc16218)

[1.2项目意义 2](#_Toc14614)

[1.3项目环境 3](#_Toc13704)

[1.3.1开发环境与工具 3](#_Toc4276)

[1.3.2运行环境 3](#_Toc9561)

[2项目需求 3](#_Toc28125)

[2.1数据要求 3](#_Toc22054)

[2.2实现流程 4](#_Toc6205)

[2.3关键核心技术 4](#_Toc24536)

[3数据采集与存储 9](#_Toc9046)

[3.1数据采集 10](#_Toc32284)

[3.2数据清洗 13](#_Toc24949)

[3.3数据保存 14](#_Toc29214)

[4数据清洗 16](#_Toc3834)

[4.1数据清洗的实现 16](#_Toc3855)

[4.2关键代码 17](#_Toc319)

[5数据可视化 17](#_Toc12305)

[5.1数据可视化实现原理 17](#_Toc15719)

[5.2数据可视化实现的步骤 18](#_Toc2612)

[6项目总结 31](#_Toc16079)

[6.1项目流程 32](#_Toc9129)

[6.2项目难点 33](#_Toc12805)

[7参考文献 36](#_Toc16738)

## **1项目概述**

## **1.1背景介绍**

在中国商业界，“胡润百富榜”是一个真正意义上的排行榜。是数十年商业史重要而有趣的一个观象仪。胡润百富榜最初的两年（1999－2000）由已故的国家副主席“[红色资本家](https://baike.baidu.com/item/%E7%BA%A2%E8%89%B2%E8%B5%84%E6%9C%AC%E5%AE%B6/7401260?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)”荣毅仁蝉联；2001年从事农业的[刘永行](https://baike.baidu.com/item/%E5%88%98%E6%B0%B8%E8%A1%8C/19769?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)、[刘永好](https://baike.baidu.com/item/%E5%88%98%E6%B0%B8%E5%A5%BD/2674639?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)兄弟问鼎首富；2002年首富是荣毅仁的儿子[荣智健](https://baike.baidu.com/item/%E8%8D%A3%E6%99%BA%E5%81%A5/805992?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)、2004－2005年从事零售的[黄光裕](https://baike.baidu.com/item/%E9%BB%84%E5%85%89%E8%A3%95/216703?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)蝉联了首富。

创建者：胡润（Rupert Hoogewerf）

1970年出生于[卢森堡](https://baike.baidu.com/item/%E5%8D%A2%E6%A3%AE%E5%A0%A1/418915?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)，1993年毕业于英国[杜伦大学](https://baike.baidu.com/item/%E6%9D%9C%E4%BC%A6%E5%A4%A7%E5%AD%A6/5400122?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)，曾于[中国人民大学](https://baike.baidu.com/item/%E4%B8%AD%E5%9B%BD%E4%BA%BA%E6%B0%91%E5%A4%A7%E5%AD%A6/138747?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)学习汉语，通晓德语、法语、[卢森堡语](https://baike.baidu.com/item/%E5%8D%A2%E6%A3%AE%E5%A0%A1%E8%AF%AD/7625675?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)、[葡萄牙语](https://baike.baidu.com/item/%E8%91%A1%E8%90%84%E7%89%99%E8%AF%AD/676611?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)、日语等七种语言。[英国注册会计师](https://baike.baidu.com/item/%E8%8B%B1%E5%9B%BD%E6%B3%A8%E5%86%8C%E4%BC%9A%E8%AE%A1%E5%B8%88/6023745?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)，有七年安达信[伦敦](https://baike.baidu.com/item/%E4%BC%A6%E6%95%A6/862?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)分部和上海分部的工作经验。

1999年创立“胡润百富榜”，2008年10月百富榜十周年。百富榜、慈善榜等一系列榜单，对于记录[中国经济](https://baike.baidu.com/item/%E4%B8%AD%E5%9B%BD%E7%BB%8F%E6%B5%8E/3049?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)进程的作用被广泛肯定。

2002年，胡润被《[新周刊](https://baike.baidu.com/item/%E6%96%B0%E5%91%A8%E5%88%8A/215397?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)》评为“年度新锐人物”。2005年，胡润首创百富榜之举被《[中国企业家](https://baike.baidu.com/item/%E4%B8%AD%E5%9B%BD%E4%BC%81%E4%B8%9A%E5%AE%B6/9486537?fromModule=lemma_inlink" \t "https://baike.baidu.com/item/%E8%83%A1%E6%B6%A6%E7%99%BE%E5%AF%8C%E6%A6%9C/_blank)》评为“改变中国商业史的20个关键时刻之一”。

## **1.2项目意义**

百富榜的出现无疑是中国经济的一个前进。百福榜让原本不为人知的富豪故事流传开来，让财富在中国不再显得那么神秘。而且，我们仍需要从不同角度对百福榜的现实意义和影响进行分析。

对富豪 胡润从推出富豪榜至今，就不断有富豪拒绝上榜，甚至有人不惜走后门托人“说情”。或许，中国人骨子里就渗透着“财不外漏”的特性。各有富豪担心上了百富榜成为“明星”后，成为腐败或逃税调查的目标，怕人盯着“承受”不起。

对大众 自从胡润百福榜退出后，普通老百姓也逐步开始了解中国富豪经济。之前，大众普遍有一种“仇富”心理，认为，这些富豪是靠其垄断北京，倒买倒卖而发家致富的。然而当富豪的创业故事随百福榜一同登出，大众的心理则有所改变。他们认识到许多民营企业家是在富民政策的指导下，靠辛勤劳动和合法经营富裕起来的，他们在创造自己富裕生活的同时，也为社会创造了财富，提供了众多就业机会。大众对富豪的看法从“敌视”转变为“崇尚”。

对政府 上榜的富豪们对于地方政府可以说是一种荣耀

## **1.3项目环境**

**1.3.1开发环境与工具**

开发工具为PyCharm 2023.1用于爬虫系统的开发，可视化系统、数据清洗，爬虫模块的开发。

可视化工具，Highcharts 库附带了您所有的工具 需要创建可靠且安全的数据可视化。基于 JavaScript 和 TypeScript 构建， 我们所有的图表库都适用于任何后端数据库或服务器堆栈。我们提供 最流行编程的包装器 语言（.Net、PHP、Python、R、Java）以及 [iOS](https://www.highcharts.com/blog/products/ios/) 和 [Android](https://www.highcharts.com/blog/products/android/) 以及框架 比如 Angular、Vue 和 React。

网络服务搭建，服务搭建使用python语言下的django和djangorestframework实现交互式可视化系统展示。

数据清洗，Pandas 是Python的数据处理包，全名：Python Data Analysis Library，Pandas 是非常著名的开源数据处理库，我们可以通过它完成对数据集进行快速读取、转换、过滤、分析等一系列操作。Pandas库是基于Numpy库来创建的，Numpy主要用于矩阵操作，而Pandas主要用于数据处理。

**1.3.2运行环境**

运行环境表1-1

|  |  |
| --- | --- |
| CPU | Intel i7 |
| 内存 | 32 |
| 硬盘 | 4T |
| 操作系统 | Windows10/Linux |

## **2项目需求**

## **2.1数据要求**

基本数据要求

1. 获取富豪的基本数据，包括：年龄，姓名，所属行业，地区，上榜时间，性别，企业名称，财富价值（美元），排名，排名变化等作为基础数据。
2. 通过pandas进行基础数据清洗，将缺失值，异常值清洗替换为所需正常值范围。
3. 将基础数据中的地区分布，通过pandas和Highcharts进行地图的制作，体现出富豪的地区分布情况。
4. 将基础数据中的排名情况，用曲线图进行展示，能体现出财富变化情况。
5. 将基础数据中的财富值，制作成3D柱状图，体现财富的变化情况
6. 将基础数据中的行业情况，制作3D饼图，体现各个行业中富豪的分布情况
7. 将基础数据中的年龄数据和财富值，制作双折线图
8. 将基础数据中的性别情况，制作成二维饼图

## **2.2实现流程**

步骤1：数据采集，通过python的requests库，及json模块对采集数据进行组装

步骤2：数据保存采用django自带sqllite数据库作为存储设备

步骤3：通过pandas模块进行数据清洗，数据分析，最终通过json数据格式返回前端，以供渲染使用。

## **2.3关键核心技术**

**2.3.1数据采集和分析技术**

该项目以python语言作为主要开发语言，Python是一种代表简单主义思想的语言。阅读一个良好的Python程序就感觉像是在读英语一样，尽管这个英语的要求非常严格！Python的这种伪代码本质是它最大的优点之一。它使你能够专注于解决问题而不是去搞明白语言本身。

Requests库简介：Requests是用python语言基于urllib编写的，采用的是Apache2 Licensed开源协议的HTTP库。

Python简介：Python是一种面向对象的解释型计算机程序设计语言，具有丰富和强大的库。它常被昵称为胶水语言，能够把用其他语言制作的各种模块（尤其是C/C++）很轻松地联结在一起。常见的一种应用情形是，使用Python快速生成程序的原型（有时甚至是程序的最终界面），然后对其中有特别要求的部分，用更合适的语言改写，比如3D游戏中的图形渲染模块，性能要求特别高，就可以用C/C++重写，而后封装为Python可以调用的扩展类库。需要注意的是在您使用扩展类库时可能需要考虑平台问题，某些可能不提供跨平台的实现。

由于Python语言的简洁性、易读性以及可扩展性，在国外用Python做科学计算的研究机构日益增多，一些知名大学已经采用Python来教授程序设计课程。例如卡耐基梅隆大学的编程基础、麻省理工学院的计算机科学及编程导论就使用Python语言讲授。众多开源的科学计算软件包都提供了Python的调用接口，例如著名的计算机视觉库OpenCV、三维可视化库VTK、医学图像处理库ITK。

而Python专用的科学计算扩展库就更多了，例如如下3个十分经典的科学计算扩展库：NumPy、SciPy和matplotlib，它们分别为Python提供了快速数组处理、数值运算以及绘图功能。因此Python语言及其众多的扩展库所构成的开发环境十分适合工程技术、科研人员处理实验数据、制作图表，甚至开发科学计算应用程序。

Django框架简介：Django是Python代码所编写开源Web框架，它具有完整的架站能力，通过该框架可以通过较少的代码片段和较为轻松的方式搭建Web应用，它问世于2003年，在2006年加入了BSD许可证（使用了BSD许可证，意味着开发者可以对软件进行任意处理，只需要在软件中注明其是来自哪里的项目即可），Django成为开源的Web应用框架。

Django具备以下特点：

1.功能完善，易上手，开发速度快，安全性强。

2.有完善的在线文档。

3.其模式自带数据库ORM（Object-Relation Mapping，对象映射）组件，使得开发者无需学习其他数据库访问技术（如SQL Alchemy等）。

4.可以使用正则表达式管理路由映射，方便灵活。

Requests 允许你发送 HTTP/1.1 请求，无需手工劳动。你不需要手动为 URL 添加查询字串，也不需要对 POST 数据进行表单编码。Keep-alive 和 HTTP 连接池的功能是 100% 自动化的，一切动力都来自根植在 Requests 内部的 urllib3。

Pandas简介： Pandas是一个开源的[Python库](https://so.csdn.net/so/search?q=Python%E5%BA%93&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_46121540/article/details/_blank)，提供快速、灵活和富有表现力的数据结构，设计目的是使得关系或标记数据的操作既简单又直观。Pandas最初被设计用于金融数据分析和数据清洗任务，但现在已被广泛应用于各种数据分析和数据处理场景。

Pandas的主要数据结构是Series（一维数据）与DataFrame（二维数据），这两种数据结构足以处理金融、统计、社会科学、工程等领域里的大多数典型的用例。对于R用户，DataFrame提供了比R语言data.frame更丰富的功能。Pandas是基于NumPy开发，可以与其他第三方科学计算支持库完美集成。

Pandas具有以下特点：

处理浮点与非浮点数据里的缺失数据，表示为NaN;

大小可变：插入或删除DataFrame等多个对象的列；

自动、显示数据对齐：显示将对象与一组标签对齐，也可以忽略标签，在Series、DataFrame计算时自动与数据对齐；

强大、灵活的分组（group by）功能：拆分－应用－组合数据集，聚合、转换数据；

把Python和NumPy数据结构里不规则、不同索引的数据轻松地转换为DataFrame对象；

基于智能标签，对大型数据集进行切片、花式索引、子集分解等操作；

直观地合并、连接数据集；

灵活地重构、透视数据集；

轴支持结构化标签：一个刻度支持多个标签；

成熟地IO工具：读取文本文件（CSV等支持分隔符地文件）、Excel文件、数据库等来源地数据，利用超快地HDF5格式保存/加载数据；

时间序列：支持日期范围生成、频率转换、移动窗口统计、移动窗口线性回归、日期位移等时间序列功能。

可视化工具eacharts简介：

ECharts，缩写来自Enterprise Charts，商业级数据图表，一个纯Javascript的图表库，可以流畅地运行在PC和移动设备上，兼容当前绝大部分浏览器（IE6/7/8/9 /10/11，chrome，firefox，Safari等），底层依赖轻量级的Canvas类库ZRender，提供直观，生动，可交互，可高度个性化定制的数据可视化图表。创新的拖拽重计算、数据视图、值域漫游等特性极大增强了用户体验，赋予了用户对数据进行挖掘、整合的能力。

支持折线图（区域图）、柱状图（条状图）、散点图（气泡图）、K线图、饼图（环形图）、雷达图（填充雷达 图）、和弦图、力导向布局图、地图、仪表盘、漏斗图、事件河流图等12类图表，同时提供标题，详情气泡、图例、值域、数据区域、时间轴、工具箱等7个可交 互组件，支持多图表、组件的联动和混搭展现。

ECharts具有以下特点：

1.可视化类型丰富，并且提供了吸引眼球的特效

2.多渲染方案，能够跨平台使用，支持以 Canvas、SVG(4.0+)、VML 的形式渲染图表。

3.多维数据的支持以及丰富的视觉编码手段，例如 对于传统的散点图等，传入的数据也可以是多个维度的。

4.多种数据格式无需转换直接使用，内置的 dataset 属性（4.0+）支持直接传入包括二维表，key-value 等多种格式的数据源，此外还支持输入 TypedArray 格式的数据。

5.无障碍访问（4.0+），能够支持自动根据图表配置项智能生成描述，使得盲人可以在朗读设备的帮助下了解图表内容，让图表可以被更多人群访问！

Sqllite简介：SQLite是一款轻型的数据库，是遵守ACID的关系型数据库管理系统，它包含在一个相对小的C库中，实现了自给自足的、无服务器的、零配置的、事务性的 SQL 数据库引擎。就像其他数据库，SQLite 引擎不是一个独立的进程，可以按应用程序需求进行静态或动态连接。SQLite 直接访问其存储文件。

模块增加和删除，在Django框架中，由于使用到了观察者模式（python中一种开发模式），使得模块的增加与删除的功能实现相对容易，这也为日后项目扩展提供了便利条件，由于数据可视化分析中，具有很多的不确定性因素，故在开发过程中应注意数据源的变化，对于数据源变化所产生的问题，在Django中，只需增加相应的应用即可，通过MVC模式可方便地实现各模块的变动。

Spider类，通过Django中orm模型映射关系，将爬虫管理模块中关于爬虫名称，爬取字段，爬取年份等相关信息，映射到Python类中，从而实现灵活处理数据库中SQL语句冗长难以记忆的缺点。

在爬虫中，通过MVC模式，实现控制爬虫与爬取内容的关联效果，从而将以往的尽在后端运行的爬虫，可以有效地迁移至前端，提供用户输入与选择接口，实现只需前端页面进行点击等相应操作，便可实现爬虫的爬取功能，提高用户体验度，提升用户交互功能。

**2.3.2网络服务器搭建**

该项目采用django和djangorestframework作为主要服务器搭建框架。

Django的主要目的是简便、快速地开发数据库驱动的网站。它强调代码复用，多个组件可以很方便地以插件形式服务于整个框架，Django有许多功能强大的第三方插件，你甚至可以很方便地开发出自己的工具包。这使得Django具有很强的可扩展性。它还强调快速开发和DRY(DoNotRepeatYourself）原则。

Django REST framework 框架是一个用于构建Web API 的强大而又灵活的工具。

通常简称为DRF框架 或 REST framework。

DRF框架是建立在Django框架基础之上，由Tom Christie大牛二次开发的开源项目。

特点：

提供了定义序列化器Serializer的方法，可以快速根据 Django ORM 或者其他库自动序列化/反序列化；

提供了丰富的类视图、Mixin扩展类，简化视图的编写；

丰富的定制层级：函数视图、类视图、视图集合到自动生成 API，满足各种需要；

多种身份认证和权限认证方式的支持；

内置了限流系统；

直观的 API web 界面；

可扩展性，插件丰富

**2.3.3可视化系统搭建**

该系统可视化部分使用html，JavaScript，css， jQuery等相关技术实现可视化系统搭建。

Html简介：HTML 英文全称是 [Hyper](https://so.csdn.net/so/search?q=Hyper&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/ccc369639963/article/details/_blank) Text Markup Language，中文译为“超文本标记语言”，专门用来设计和编辑网页。

JavaScript简介：JavaScript一种解释性[脚本语言](https://so.csdn.net/so/search?q=%E8%84%9A%E6%9C%AC%E8%AF%AD%E8%A8%80&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_60934893/article/details/_blank)，是一种动态类型、弱类型、基于原型继承的语言，内置支持类型。它的[解释器](https://so.csdn.net/so/search?q=%E8%A7%A3%E9%87%8A%E5%99%A8&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/weixin_60934893/article/details/_blank)被称为JavaScript引擎，作为浏览器的一部分，广泛用于客户端的脚本语言，用来给HTML网页增加动态功能。

Css简介：CSS也叫做层迭样式表，它是一种定义样式结构如字体、颜色、位置等的语言，对 HTML 代码进行美化。

jQuery简介：jQuery是一个快速，小型且功能丰富的JavaScript库。借助易于使用的API（可在多种浏览器中使用），使HTML文档的遍历和操作，事件处理，动画和Ajax等事情变得更加简单。兼具多功能性和可扩展性，jQuery改变了数百万人编写JavaScript的方式。兼容多浏览器的javascript函数库，把我们常用的一些功能进行了封装，方便我们来调用，提高我们的开发效率。核心理念是写得更少，做得更多（write less,do more）。

可视化图表制作工具Highcharts简介：Highcharts 是一个用纯 JavaScript 编写的一个图表库， 能够很简单便捷地在 Web 网站或是 Web 应用程序添加有交互性的图表，并且免费提供给个人学习、个人网站和非商业用途使用。Highcharts 支持的图表类型有直线图、曲线图、区域图、柱状图、饼状图、散状点图、仪表图、气泡图、瀑布流图等多达 20 种图表，其中很多图表可以集成在同一个图形中形成混合图。

可视化大屏制作，采用JavaScript中的echarts技术，通过ajax请求向后端发送请求，在HTML中使用JavaScript实现数据渲染。

**2.3.3系统测试**

系统开发是一个无法确保程序完全正确的过程，也不能避免出现 BUG，但是不能让这些 BUG时时刻刻都在运行，一个 BUG就会给整个系统带来不可挽回的损失，比如：系统崩溃、安全信息泄漏、系统故障等。要防止不断出现的BUG，就必须对整个系统进行软件测试，在测试中发现问题，并及时修复，只有这样才能使整个系统更加牢固、更加完善。

在程序设计完成后，对系统进行测试是必须的，一般采用软件测试法。在新的软件开发阶段，系统测试是对软件品质进行检测一个重要环节，同时也是满足设计目的和是否符合预期的一个重要依据。测试的重点在于检查软件中的数据和动作的精确度，以及运行结果的正确性，以及需要提高的地方。

系统进行测试时，一旦发现了问题就必须在最短时间内尽早想办法解决，只有这样才能让系统顺利高质量地完成，并且可以大幅度地减少开发的时间。在进行测试的过程中，要尽量避免出现相同或类似的错误，每遇到一个错误问题，就要将所有系统开发中所涉及的同类问题全部解决，只有这样才能保证系统的可靠性。

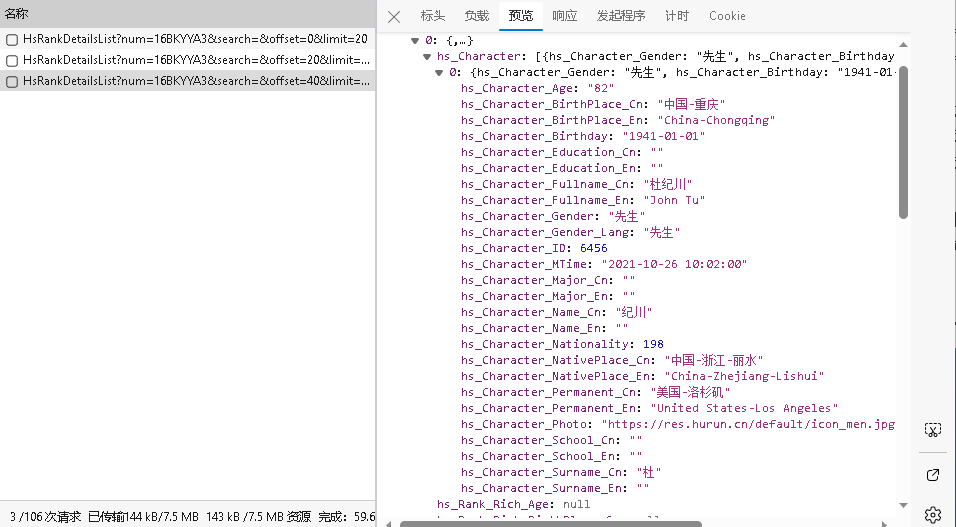
**3数据采集与存储**

## **3.1数据采集**

该网页链接地址为https://www.hurun.net/数据采集前需对网页进行必要的分析。

1. 通过对网页初步分析，发现该数据加载方式为ajax动态加载，通过F12进入开发者工具刷新页面，发现数据由该接口https://www.hurun.net/zh-CN/Rank/HsRankDetailsList?num=16BKYYA3&search=&offset=40&limit=20返回

图3.1（接口数据分析）



由于该数据返回格式为json格式，故使用python中的json模块进行数据解析。

1. 通过翻页发现该接口实现翻页的参数为offset参数，每页需要增加offset的值为limit的整倍数，故翻页逻辑可通过该参数进行实现，以便获取更多数据

图3.2



1. 通过python中的json模块将采集到的数据组装成便于存储的格式，并进行必要 的数据筛选工作，如下：

表3.3（数据返回解析代码）

|  |
| --- |
| "hs\_Character":                "hs\_Character\_Gender": 先生",              "hs\_Character\_Birthday": "1941-01-01",              "hs\_Character\_Age": "82",              "hs\_Character\_MTime": "2021-10-26 10:02:00",              "hs\_Character\_Gender\_Lang": "先生",              "hs\_Character\_ID": 6456,              "hs\_Character\_Fullname\_Cn": "杜纪川",              "hs\_Character\_Fullname\_En": "John Tu",              "hs\_Character\_Surname\_Cn": "杜",              "hs\_Character\_Name\_Cn": "纪川",              "hs\_Character\_Surname\_En": "",              "hs\_Character\_Name\_En": "",              "hs\_Character\_Photo": "https://res.hurun.cn/default/icon\_me n.jpg",              "hs\_Character\_Nationality": 198,              "hs\_Character\_NativePlace\_Cn": "中国-浙江-丽水",              "hs\_Character\_NativePlace\_En": "China-Zhejiang-Lishui",              "hs\_Character\_BirthPlace\_Cn": "中国-重庆",              "hs\_Character\_BirthPlace\_En": "China-Chongqing",              "hs\_Character\_Permanent\_Cn": "美国-洛杉矶",              "hs\_Character\_Permanent\_En": "United States-Los Angeles",              "hs\_Character\_Education\_Cn": "",              "hs\_Character\_Education\_En": "",              "hs\_Character\_School\_Cn": "",              "hs\_Character\_School\_En": "",              "hs\_Character\_Major\_Cn": "",              "hs\_Character\_Major\_En": ""           }       ],       "hs\_Rank\_Rich\_Nationality": null,       "hs\_Rank\_Rich\_NativePlace\_Cn": null,       "hs\_Rank\_Rich\_NativePlace\_En": null,       "hs\_Rank\_Rich\_BirthPlace\_Cn": null,       "hs\_Rank\_Rich\_BirthPlace\_En": null,       "hs\_Rank\_Rich\_Permanent\_Cn": null,       "hs\_Rank\_Rich\_Permanent\_En": null,       "hs\_Rank\_Rich\_Photo": null,       "hs\_Rank\_Rich\_Age": null,       "hs\_Rank\_Rich\_Gender": null,       "hs\_Rank\_Rich\_Education\_Cn": null,       "hs\_Rank\_Rich\_Education\_En": null,       "hs\_Rank\_Rich\_School\_Cn": null,       "hs\_Rank\_Rich\_School\_En": null,       "hs\_Rank\_Rich\_Year": "2023",       "hs\_Rank\_Rich\_Relations": "未知",       "hs\_Rank\_Rich\_MTime": "2023-10-24 12:00:00",       "hs\_Rank\_Rich\_ID": 26353,       "hs\_Rank\_Rich\_ListID": 141,       "hs\_Rank\_Rich\_Ranking": 40,       "hs\_Rank\_Rich\_Ranking\_Change": "54",       "hs\_Rank\_Rich\_Wealth": 800,       "hs\_Rank\_Rich\_Wealth\_USD": 11000,       "hs\_Rank\_Rich\_Wealth\_Change": "63%",       "hs\_Rank\_Rich\_Certificate": null,       "hs\_Rank\_Rich\_ChaName\_Cn": "杜纪川",       "hs\_Rank\_Rich\_ChaName\_En": "John Tu",       "hs\_Rank\_Rich\_Cha\_ID": "6456",       "hs\_Rank\_Rich\_ComName\_Cn": "金士顿科技",       "hs\_Rank\_Rich\_ComName\_En": "Kingston Technology",       "hs\_Rank\_Rich\_ComHeadquarters\_Cn": "美国-洛杉矶",       "hs\_Rank\_Rich\_ComHeadquarters\_En": "USA-Los Angeles",       "hs\_Rank\_Rich\_Industry\_Cn": "半导体",       "hs\_Rank\_Rich\_Industry\_En": "Semiconductor",       "hs\_Rank\_Rich\_Com\_ID": "0",       "hs\_RichAndCha": null  } |

1. 通过django中自带的sqllite3及model模块，实现数据的入库工作。

**3.2数据清洗**

该系统数据清洗模块采用pandas做基础数据清洗工作，首先去除数据中的空值和异常值，将数据处理为可视化所需要的数据格式，通过Highcharts模块实现前端页面的渲染展示

城市数据清洗表3.3

|  |
| --- |
| def city\_charts(self):  data = self.get\_data()  citys = data['省份'].value\_counts()  provinces = ["北京", "天津", "上海", "重庆", "新疆", "西藏", "宁夏", "内蒙古","广西", "黑龙江", "吉林", "辽宁", "河北", "山东", "江苏", "安徽","浙江", "福建", "广东", "海南", "云南", "贵州", "四川", "湖南","湖北", "河南", "山西", "陕西", "甘肃", "青海", "江西", "台湾", "香港", "澳门"]  json\_data = {}  for pro in provinces:  city\_count = citys.get(pro)  json\_data[pro] = city\_count  return json\_data |

去除空值和异常值表3.4

|  |
| --- |
| def shop\_buy(self):  dataframe = self.df  new\_df = dataframe.dropna()  df1 = new\_df.iloc[:, [1, 3, 4]]  df2 = df1.str.split(' ', expand=True)  df3 = pandas.concat([df1, df2], axis=1).values  data = []  i = 0  for df\_info in df3:  i += 1  df = {  "id": i,  "sale\_date": df\_info[3],  "sale\_time": df\_info[4],  "sale\_price": df\_info[0],  "aero": df\_info[1]  }  data.append(df)  return data |

**3.3数据保存**

该系统采用django自带sqlite3作为存储载体，数据清洗结束，通过调用模型匹配相应字段，使用djangorestframework的序列化器进行反序列化，实现最终的保存入库

表3.5（数据模型创建）

|  |
| --- |
| class DataModel(models.Model):  Gender = models.CharField(verbose\_name='性别', max\_length=128, null=True, blank=True)  Age = models.CharField(verbose\_name='出生日期', max\_length=128, null=True, blank=True)  Fullname = models.CharField(verbose\_name='姓名', max\_length=128, null=True, blank=True)  NativePlace = models.CharField(verbose\_name='地区', max\_length=128, null=True, blank=True)  BirthPlace = models.CharField(verbose\_name='籍贯', max\_length=128, null=True, blank=True)  Permanent = models.CharField(verbose\_name='现居住地', max\_length=128, null=True, blank=True)  Year = models.CharField(verbose\_name='年份', max\_length=128, null=True, blank=True)  Industry = models.CharField(verbose\_name='行业', max\_length=128, null=True, blank=True)  ComName = models.CharField(verbose\_name='公司名称', max\_length=128, null=True, blank=True)  Ranking = models.CharField(verbose\_name='排名', max\_length=128, null=True, blank=True)  Wealth = models.CharField(verbose\_name='财富值', max\_length=128, null=True, blank=True)  Change = models.CharField(verbose\_name='排名变化', max\_length=128, null=True, blank=True)  Wealth\_Change = models.CharField(verbose\_name='变化率', max\_length=128, null=True, blank=True)  update\_time = models.DateTimeField(verbose\_name='更新时间', auto\_now\_add=True) |

|  |
| --- |
| class DtaSerializers(ModelSerializer):  class Meta:  model = DataModel  fields = '\_\_all\_\_' |

表3.6（数据模型序列化器）

## **4数据清洗**

**4.1数据清洗的实现**

数据读取通过pandas模块对已下载数据进行读取，并将元数据修改为可进行分析和可视化视图的格式类型，对原始数据中存在的异常值以及空数据进行替换和统一类型的操作。

相关字段的整理与统计，通过dataframe类型数据进行相应字段的提取与转换，数据清洗通过专用的cleaning模块完成，该模块中包含可视化数据的整理与形成，其中包括地图形成，柱状图形成，饼图等其他可视化视图的形成。

表4.1（数据清洗核心代码）

|  |
| --- |
| def to\_dataframe(self):  dataframe = pandas.DataFrame(self.data, columns=['hs\_Rank\_Rich\_ChaName\_Cn',  'hs\_Character\_Gender',  'hs\_Character\_Age',  'hs\_Rank\_Rich\_ComHeadquarters\_Cn',  'hs\_Rank\_Rich\_Ranking',  'hs\_Rank\_Rich\_Relations',  'hs\_Rank\_Rich\_Wealth',  'hs\_Rank\_Rich\_Year',  'hs\_Rank\_Rich\_Ranking\_Change',  'hs\_Rank\_Rich\_ComName\_Cn',  'hs\_Rank\_Rich\_Industry\_Cn'])  return dataframe |

通过清洗代码对原始数据的二次处理，展示在前端页面中，并将数据展示为便于理解的视图形式，其中对结果的保存，通过eacharts中的到处模块实现，并保存到对应文件夹中，以便后期进行处理和持久化保存。

系统中对可视化图标的保存提供了直接打印，保存图片格式等多种方式，便于用户选择适合自己的保存格式。

系统中对于数据的保存提供了直接打印和保存位文件的方式，保存文件其中有csv和Excel两种下载保存的格式。

图4.2（数据展示图）



**4.2关键代码**

数据清洗关键代码与可视化代码如下表：

表4.3（数据清洗核心代码）

|  |
| --- |
| def three(self):  data\_frame = self.to\_dataframe()  dataframe = self.to\_dataframe()  data=dataframe[dataframe['hs\_Rank\_Rich\_Year'] '2023']['hs\_Rank\_Rich\_Relations'].value\_counts()  data\_dict = data.to\_dict()  sum\_data = 0  for value in data\_dict.values():  sum\_data += value  result = []  for key in data\_dict.keys():  v = data\_dict.get(key)  result.append([key, round((v / sum\_data) \* 100, 2)])  return result |

## **5数据可视化**

**5.1数据可视化实现原理**

ECharts是百度的一个项目，后来百度把Echart捐给apache，用于图表展示，提供了常规的折线图、柱状图、散点图、饼图、K线图，用于统计的盒形图，用于地理数据可视化的地图、热力图、线图，用于关系数据可视化的关系图、treemap、旭日图，多维数据可视化的平行坐标，还有用于 BI 的漏斗图，仪表盘，并且支持图与图之间的混搭。

**5.2数据可视化实现的步骤**

本项目中使用到python中的pandas第三方库，实现基础数据的重新计算，最终生成ECharts可视化所需要的数据格式与数据类型，其中传输通过ajax技术实现异步传输，做到数据的实时动态更新，Django代码中通过djangorestframework第三方库实现json数据的返回，以便提升传输效率，提高用户体验度。

表5.1（数据传输核心代码）

|  |
| --- |
| def list(self, request, \*args, \*\*kwargs):  data = self.queryset  word = request.query\_params  print(word.get('key'))  if word.get('key') == 'city':  info = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=info)  df = [chart\_info.city()]  return Response({'code': 200, 'msg': '数据获取成功', 'data': df})  elif word.get('key') == 'score':  info = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=info)  df = chart\_info.score()  value = 0  data = []  for values in df.values():  value+=values  for k, v in df.items():  data.append([k, round(v/value\*100, 2)])  return Response({'code': 200, 'msg': '数据获取成功', 'data': data})  elif word.get('key') == 'word':  info = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=info)  data = chart\_info.word\_count()  return Response({'code': 200, 'msg': '数据获取成功', 'data': data})  elif word.get('key') == 'date':  info = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=info)  data = chart\_info.date()  return Response({'code': 200, 'msg': '数据获取成功', 'data': data})  elif word.get('key') == 'cloud':  info = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=info)  data = chart\_info.cloud()  return Response(status=200, data={'data': data})  elif word.get('key') == 'three':  info = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=info)  data = chart\_info.three()  return Response(status=200, data={'data': data}) |

表5.2（前端展示核心代码）

|  |
| --- |
| series.push(  {  //name: ' Top10',  type: 'effectScatter',//带有涟漪特效动画的散点（气泡）图  coordinateSystem: 'geo',//系列使用的坐标系,geo为地理坐标系  zlevel: 2,  rippleEffect: {//涟漪特效相关配置  brushType: 'stroke'//波纹的绘制方式，可选 'stroke' 和 'fill'。  },  label: {//标签相关配置  normal: {  show: true,  position: 'right',  text: item.count,  weight:300,  height:50,  formatter: '{b}'//{a}、{b}、{c}，分别表示系列名，数据名，数据值  },  text:item.count,  },  symbolSize: 10,  itemStyle: {  normal: {  color: '#ff0000',  weight: 300,  height: 50  },  text:item.count  },  text:item.count,  data: [item],  tooltip:{  formatter:function () {  return '上榜人数<br>数量：'+item.count  },  text: item.count,  show:true,  weight:300,  height:50  }  } |

表**4.3**

图4.4富豪榜综合数据展示图



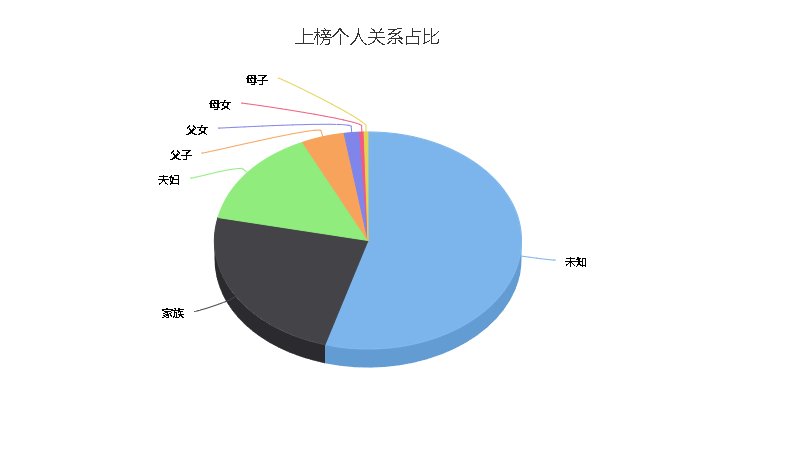
通过对数据的综合处理，最终实现更便于理解的可视化视图，包括地图，换装图，曲线图，饼图和柱状图等。

表4.5（地区分布后端核心代码）

|  |
| --- |
| def city(self):  dataframe = self.to\_dataframe()  city\_data = dataframe[dataframe['hs\_Rank\_Rich\_Ranking'].astype('int')<11]  data\_ = city\_data[['hs\_Rank\_Rich\_Year','hs\_Rank\_Rich\_Wealth']]  data\_['hs\_Rank\_Rich\_Wealth'].astype('float')  key = data\_.to\_dict().get('hs\_Rank\_Rich\_Year').values()  value = data\_.to\_dict().get('hs\_Rank\_Rich\_Wealth').values()  name = list(set(key))  year\_ = {}  for year\_\_ in name:  year\_[year\_\_] = []  # print(year\_)  for year, wealth in zip(key, value):  year\_[year].append(float(wealth))  return year\_ |

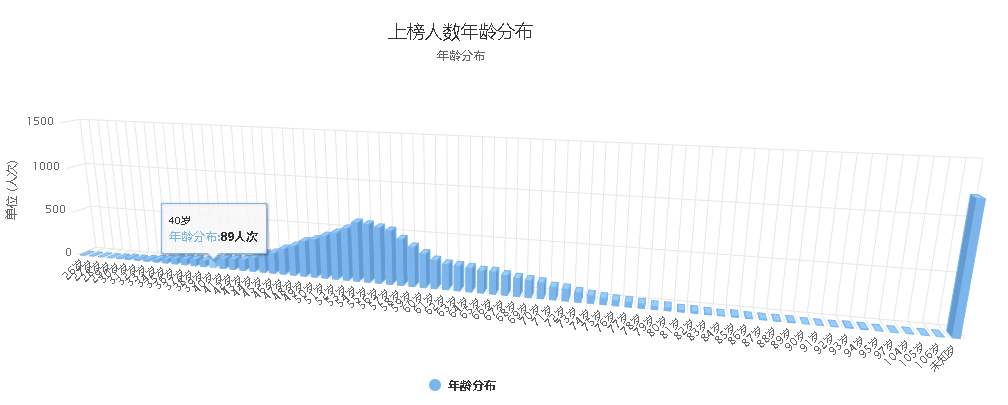
通过后 端返回到前端的包含坐标信息及展示数据，将json数据展示到前端页面，核心使用echarts模块进行数据解析和页面渲染

图4.7（家族占比图）



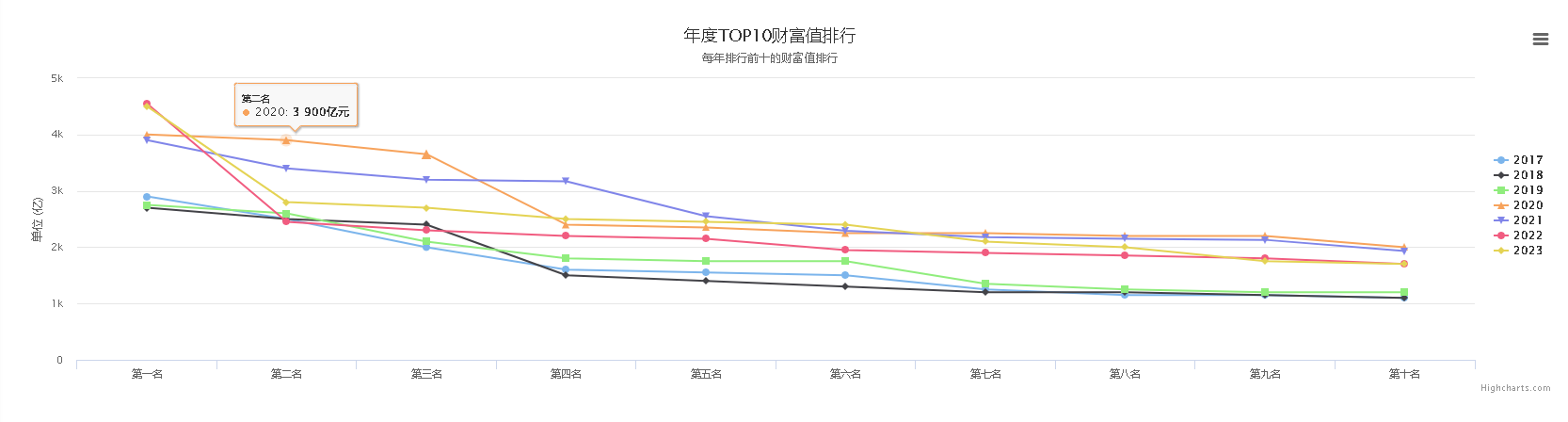
通过饼图，显示家族企业在富豪榜中的排行，体现家族企业目前的发展状况。

.图4.8（财富值柱状图）



获取到数据中的年龄维度，从何得到最集中的年龄分布情况。

图4.9（各年度前十名财富值排行）



通过近几年对上榜企业前十名的财富值统计，得到前十大企业中财富值的变化情况，对以后经济发展起到指导作用。

图4.10（上榜企业图）



分析得出上榜企业的数据，从而获得哪些企业更易积累财富值

表4.11（后端核心代码）

|  |
| --- |
| def list(self, request, \*args, \*\*kwargs):  status = {'code': 200, 'msg': '请求成功', 'data': 'data'}  key = request.query\_params.get('key', {})  data = self.queryset  if key == 'word':  data = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=data)  info = {'category': [],  'value': []}  for key, value in chart\_info.s\_word\_count().items():  info['category'].append(key)  info['value'].append(value)  status['data'] = info  return Response(status)  if key == 'city':  data = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=data)  info = {'category': [],  'value': []}  for key, value in chart\_info.s\_city().items():  info['category'].append(key)  info['value'].append(value)  status['data'] = info  return Response(status)  if key == 'aero':  data = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=data)  data\_ = []  info\_ = []  for key, value in chart\_info.s\_score().items():  data\_.append(key)  info\_.append({'value': value, 'name': key})  data\_info = {  'data': data\_,  'info': info\_  }  status['data'] = data\_info  return Response(status)  if key == 'gender':  data = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=data)  data\_ = []  info\_ = []  for key, value in chart\_info.s\_date().items():  data\_.append(key)  info\_.append({'value': value, 'name': key})  data\_info = {  'data': data\_,  'info': info\_  }  status['data'] = data\_info  return Response(status)  if key == 'date':  data = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=data)  data\_ = []  info\_ = []  for key, value in chart\_info.video().items():  data\_.append(key)  info\_.append({'value': value, 'name': key})  data\_info = {  'data': data\_,  'info': info\_  }  status['data'] = data\_info  return Response(status)  if key == 'video':  video\_data = self.serializer\_class(data, many=True)  chart\_info = Clearing(data=video\_data.data)  info = {'category': [],  'value': []}  for key, value in chart\_info.video().items():  info['category'].append(key)  info['value'].append(value)  status['data'] = info  return Response(status)  if key == 'video\_overview':  video\_data = self.serializer\_class(data, many=True)  chart\_info = Clearing(data=video\_data.data)  info = {  'category': [1,2,3,4,5,6,7,8,9,10],  'hs\_Rank\_Rich\_Ranking': list(chart\_info.video\_overview().get('hs\_Rank\_Rich\_Ranking').values()),  'hs\_Rank\_Rich\_Wealth': list(chart\_info.video\_overview().get('hs\_Rank\_Rich\_Wealth').values()),  'hs\_Character\_Age': list(chart\_info.video\_overview().get('hs\_Rank\_Rich\_Year').values()),  }  status['data'] = info  return Response(status)  if key == 's\_comment':  info = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=info)  df = chart\_info.s\_comment()  all\_count = sum(df.values())  info = []  for key, value in df.items():  score = round(value / all\_count \* 100, 2)  info\_ = {  'name': key,  'value': score  }  info.append(info\_)  status['data'] = info  if key == 'video\_city':  video\_data = self.get\_serializer(data, many=True).data  chart\_info = Clearing(data=video\_data)  info = []  for key, value in chart\_info.video\_city().items():  info.append({  'name': key,  'value': value  })  status['data'] = info  return Response(status)  if key == 'title':  video\_data = self.get\_serializer(data, many=True)  chart\_info = Clearing(data=video\_data.data)  data = chart\_info.screen()  # info = {'category': [],  # 'value': []}  # for key, value in chart\_info.video().items():  # info['category'].append(key)  # info['value'].append(value)  status['data'] = data  return Response(status)  return Response(status) |

表4.12（前端核心代码）

|  |
| --- |
| <!doctype html>  <html>  <head>  <meta charset="utf-8">  <title>index</title>  <script type="text/javascript" src="../../static/screen/js/jquery.js"></script>  <link rel="stylesheet" href="../../static/screen/css/comon0.css">  </head>  <script>  $(window).load(function(){  $(".loading").fadeOut()  })    /\*\*\*\*/  $(document).ready(function(){  var whei=$(window).width()  $("html").css({fontSize:whei/20})  $(window).resize(function(){  var whei=$(window).width()  $("html").css({fontSize:whei/20})  });  });  </script>  <script type="text/javascript" src="../../static/screen/js/echarts.min.js"></script>  <script language="JavaScript" src="../../static/screen/js/js.js"></script>  <body>  <div class="canvas" style="opacity: .2">  <iframe frameborder="0" src="../../static/screen/js/index.html" style="width: 100%; height: 100%"></iframe>  </div>  <div class="loading">  <div class="loadbox"> <img src="../../static/screen/picture/loading.gif"> 页面加载中... </div>  </div>  <div class="head">  <h1>胡润富豪数据可视化大屏</h1>  <div class="weather"><!--<img src="picture/weather.png"><span>多云转小雨</span>--><span id="showTime"></span></div>    <script>  var t = null;  t = setTimeout(time,1000);//開始运行  function time()  {  clearTimeout(t);//清除定时器  dt = new Date();  var y=dt.getFullYear();  var mt=dt.getMonth()+1;  var day=dt.getDate();  var h=dt.getHours();//获取时  var m=dt.getMinutes();//获取分  var s=dt.getSeconds();//获取秒  document.getElementById("showTime").innerHTML = y+"年"+mt+"月"+day+"-"+h+"时"+m+"分"+s+"秒";  t = setTimeout(time,1000); //设定定时器，循环运行  }  </script>      </div>  <div class="mainbox">  <ul class="clearfix">  <li>  <div class="boxall" style="height: 3.2rem">  <div class="alltitle">十大上榜企业</div>  <div class="allnav" id="echart1"></div>  <div class="boxfoot"></div>  </div>  <div class="boxall" style="height: 3.2rem">  <div class="alltitle">十大上榜地区</div>  <div class="allnav" id="echart2"></div>  <div class="boxfoot"></div>  </div>  <div class="boxall" style="height: 3.2rem">  <div style="height:100%; width: 100%;">  <div class="sy" id="fb1"></div>  <div class="sy" id="fb2"></div>  <div class="sy" id="fb3"></div>  </div>  <div class="boxfoot">    </div>  </div>  </li>  <li>  <div class="bar">  <div class="barbox">  <ul class="clearfix">  <li class="pulll\_left counter" id="reply">12581189</li>  <li class="pulll\_left counter" id="danmaku">3912410</li>  </ul>  </div>  <div class="barbox2">  <ul class="clearfix">  <li class="pulll\_left">上榜企业数 </li>  <li class="pulll\_left">总财富值</li>  </ul>  </div>  </div>  <div class="map">  <div class="map1"><img src="../../static/screen/picture/lbx.png"></div>  <div class="map2"><img src="../../static/screen/picture/jt.png"></div>  <div class="map3"><img src="../../static/screen/picture/map.png"></div>  <div class="map4" id="map\_1"></div>  </div>  </li>  <li>  <div class="boxall" style="height:3.4rem">  <div class="alltitle">排名榜</div>  <div class="allnav" id="echart4"></div>  <div class="boxfoot"></div>  </div>  <div class="boxall" style="height: 3.2rem">  <div class="alltitle">个人排行榜</div>  <div class="allnav" id="echart5"></div>  <div class="boxfoot"></div>  </div>  <div class="boxall" style="height: 3rem">  <div class="alltitle">家族占有率</div>  <div class="allnav" id="echart6"></div>  <div class="boxfoot"></div>  </div>  </li>  </ul>  </div>  <div class="back"></div>  <script type="text/javascript" src="../../static/screen/js/china.js"></script>  <script type="text/javascript" src="../../static/screen/js/area\_echarts.js"></script>  <script type="text/javascript" src="../../static/js/layui/layui.js"></script>  </body>  </html> |

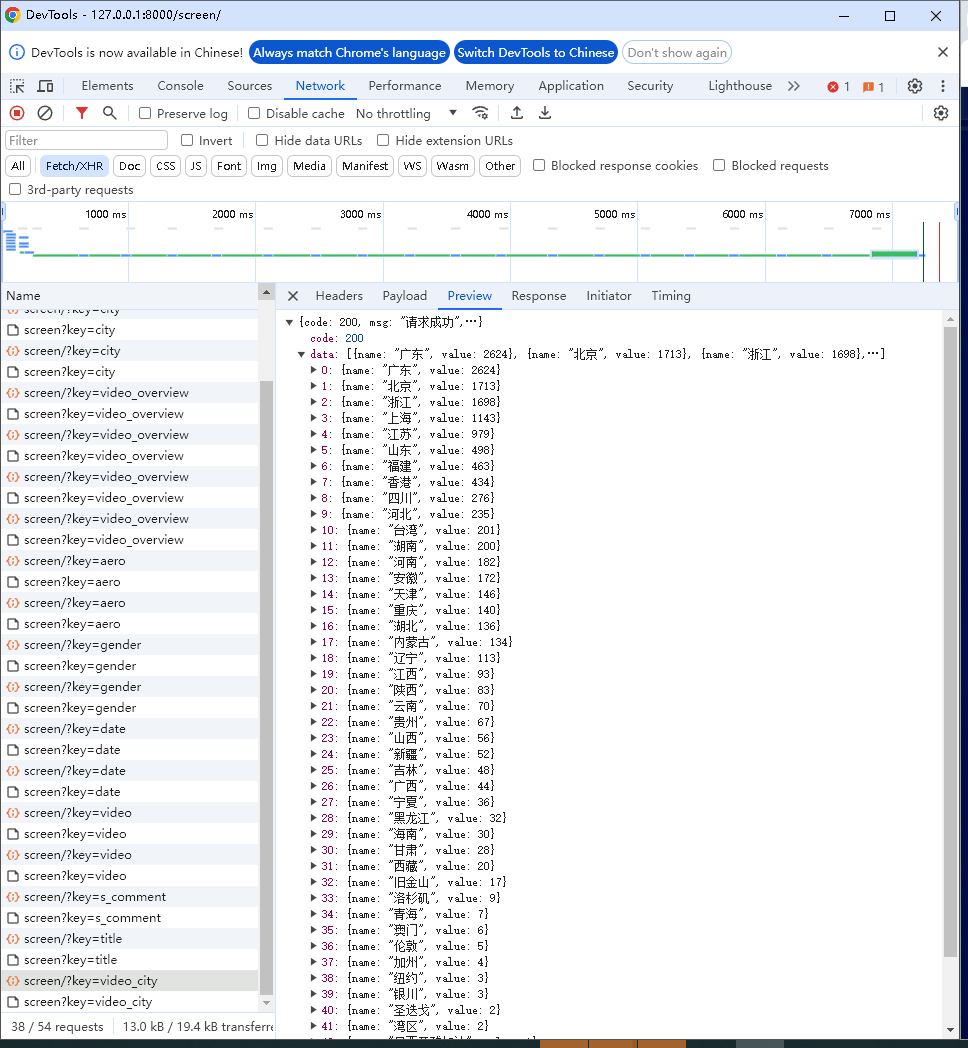
可视化大屏实现方法及核心代码：

Django在后端实现个接口数据的生成，将数据统一返回为Json数据，前端js代码通过ajax方法获取Json数据，渲染到前端页面中：

图4.13（可视化大屏预览图）



Django接口中Json数据展示



**图4.14（Json数据图）**

该项目中前后端交互均采用Json数据格式，该数据格式具有易于程序员阅读和编写。

易于计算机解析和生成。

Json数据类型特点

图4.15（Json数据类型特点）



图4.16（数据接口图）



该接口为该项目所有数据返回接口，可视化大屏仅将此接口数据集成于一个页面

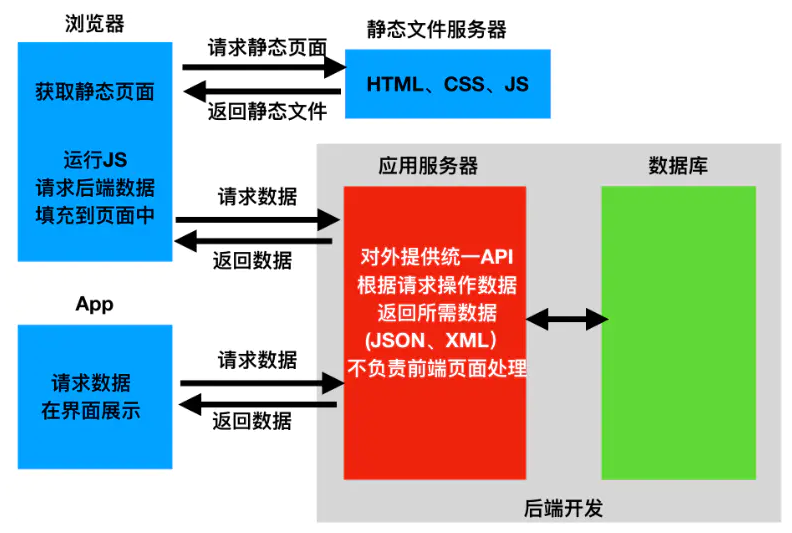
## **6项目总结**

该项目采用python开发语言为主要开发语言，网络服务使用django框架作为主框架，数据分析采用pandas为主核心数据分析，前端通过JavaScript，HTML，CSS为基础，使用轻量级的jQuery框架作为前端主要框架，结合Echarts将数据分析结果渲染于前端，达到最终效果。

其中，在该项目中，大量采用前后端分离的开发模式，该模式具有开发速度快，耦合度低的特点，在开发的各个阶段，都不相互受影响。

**6.1项目流程**

**图5.1（前后端开发模式图）**



其中，数据传输采用ajax异步传输技术，后台django借助强大的路由协同和视图系统，为每个可视图提供可靠稳定的数据。

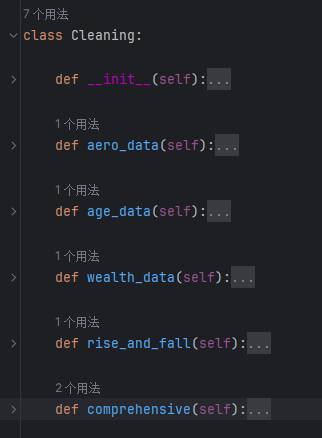
该模块负责所有前端页面的显示，实际并不参与具体数据的传输

表5.3（后端数据提供路由）

|  |
| --- |
| from .views import \*  from rest\_framework.routers import SimpleRouter  API\_router = SimpleRouter()  API\_router.register('data', DataViewSet, 'data')  API\_router.register('spider', SpiderViewSet, 'spider')  API\_router.register('aero', AeroViewSet, 'aero')  API\_router.register('age', AgeViewSet, 'age')  API\_router.register('wealth', WealthViewSet, 'wealth')  API\_router.register('rise', RiseViewSet, 'rise')  API\_router.register('comprehensive', ComprehensiveViewSet, 'comprehensive')  urlpatterns = []  urlpatterns += API\_router.urls |

该模块主要为前端提供数据分析结果数据，通过调用pandas模块实现快速数据分析。

图5.4（数据分析模块）



**6.2项目难点**

在Django项目中，前端页面模块开发完毕后，与Django路由及视图进行前后端联调时，页面样式无法加载，在多方寻找答案中，最终发现是Django配置文件中static的静态文件没有指定，导致前端无法加载css文件和js文件。

DRF开发api接口的过程中，发送ajax无法获取到Json数据，最终发现，跨域设置未设置，在解决掉跨域设置后，仍然无法获取到Json数据，最后发现由于设置了permission类，并设置为全局默认认证类，从而导致前端请求是未携带token字符串，后端响应未认证用户。

可视化大屏调试过程中，由于大量使用JavaScript语言，初期数据请求设计并不合理，导致长时间加载并无数据返回，而可视化大屏一直无法显示相应图表文件。

该项目数据存储采用sqlite3数据库，该数据库是一个 C 语言库，它实现了[一个小型](https://www.sqlite.org/footprint.html)、[快速](https://www.sqlite.org/fasterthanfs.html)、[独立](https://www.sqlite.org/selfcontained.html)、[高可靠性](https://www.sqlite.org/hirely.html)、功能[齐全](https://www.sqlite.org/fullsql.html)的 SQL 数据库引擎。 SQLite是世界上[使用最多的](https://www.sqlite.org/mostdeployed.html)数据库引擎。 SQLite内置于所有手机和大多数计算机中，并且 捆绑在无数其他应用程序中，人们 每天使用。

在获取数据库路径时，需要先获取当前文件的上级路径，在获取sqlite3数据库文件路径，由于该使用数据库文件并未与数据库在同一目录，故需要想获取上级目录，从上级目录获取数据库路径。

|  |
| --- |
| url = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))  db\_url = url + '\\db.sqlite3' |

表5.5（数据库路径获取）

前端地图在展示动态数据时，由于JavaScript渲染需要组装大量数据，故在组装动态数据时需要对后端返回的json数据进行重新解包和组装。

表5.6（前端数据组装代码）

|  |
| --- |
| {  //name: ' Top10',  type: 'effectScatter',//带有涟漪特效动画的散点（气泡）图  coordinateSystem: 'geo',//系列使用的坐标系,geo为地理坐标系  zlevel: 2,  rippleEffect: {//涟漪特效相关配置  brushType: 'stroke'//波纹的绘制方式，可选 'stroke' 和 'fill'。  },  label: {//标签相关配置  normal: {  show: true,  position: 'right',  text: item.count,  weight:300,  height:50,  formatter: '{b}'//{a}、{b}、{c}，分别表示系列名，数据名，数据值  },  text:item.count,  },  symbolSize: item.count/15+15,  itemStyle: {  normal: {  color: '#ff0000',  weight: 300,  height: 50  },  text:item.count  },  text:item.count,  data: [item],  tooltip:{  formatter:function () {  return '上榜人数<br>数量：'+item.count  },  text: item.count,  show:true,  weight:300,  height:50  }  } |

## **7参考文献**

Django官方开发文档

Django REST框架开发文档

Apache ECharts可视化图标库开发文档

jQuery框架开发文档

Sqlite3数据库开发文档