

项目文档

软件学院暑期实训项目

基于往年航班和天气信息的对航班延迟信息的预测系统

作者：实训第二小组

指导老师：鲁凌云

2022/7/16

软件学院大二学期下暑期实训

目录

[数据/算法文档 6](#_Toc108895923)

[1.数据库对接、搭建工作 6](#_Toc108895924)

[1.1搭建合适的数据库 6](#_Toc108895925)

[1.1.1概念模型 6](#_Toc108895926)

[1.1.2逻辑模型 7](#_Toc108895927)

[1.1.3物理模型 8](#_Toc108895928)

[1.1.4ER图 9](#_Toc108895929)

[1.2配合后端完成数据库的对接工作 10](#_Toc108895930)

[1.2.1数据库配置对接 10](#_Toc108895931)

[2.气象预测算法寻找数据 12](#_Toc108895932)

[3.气象预测算法数据预处理 12](#_Toc108895933)

[3.1计算均值填充缺少的数值 12](#_Toc108895934)

[4.气象预测算法的实现 12](#_Toc108895935)

[4.1使用GetData.py编写相关的爬虫程序爬取数据 12](#_Toc108895936)

[4.2通过Write.py进行爬取数据的分析、清洗；之后写到创建好的csv文件中进行下一步的数据预处理 13](#_Toc108895937)

[4.3通过ProcessData.py调用Write中的方法write进行数据的读取与csv文件的写入，并通过这些文件进行数据集的处理与划分。 14](#_Toc108895938)

[4.4最后模型构建在GetModel.py中，使用随机森林模型，填入处理好的数据集，进行模型的训练与保存 15](#_Toc108895939)

[5.航班延迟预测算法寻找数据 16](#_Toc108895940)

[5.1寻找到合适的数据集 16](#_Toc108895941)

[6.航班延迟预测算法数据预处理 17](#_Toc108895942)

[6.1缩减数据集，删除不是选中机场的出发机场和到达机场的行 17](#_Toc108895943)

[6.2删除取消的航班 18](#_Toc108895944)

[6.3将时间戳转化为普通时间 18](#_Toc108895945)

[6.4延迟时间计算 18](#_Toc108895946)

[6.5通过两地经纬度计算距离 18](#_Toc108895947)

[6.6删除多余的列 18](#_Toc108895948)

[6.7将计划出发日期转化为年，月，日，起飞时间的列 19](#_Toc108895949)

[6.8将出发延迟转化为分类预测对象 19](#_Toc108895950)

[6.9爬取所有地点的2015-2017年每天的气象数据 20](#_Toc108895951)

[6.10将气象数据整合到处理好的航班延误数据集中，合并为最终的完整数据集 21](#_Toc108895952)

[7.航班延迟预测算法的实现 22](#_Toc108895953)

[7.1数据分析 22](#_Toc108895954)

[7.1.1延迟程度频次统计图 22](#_Toc108895955)

[7.1.2相关性分析 22](#_Toc108895956)

[7.2使用基本的预测算法，成功实现基础预测功能 23](#_Toc108895957)

[7.2.1将字符型数据转化为字典映射编码 23](#_Toc108895958)

[7.2.2划分数据集 23](#_Toc108895959)

[7.2.3标准化处理 23](#_Toc108895960)

[7.2.4建模 23](#_Toc108895961)

[7.3对算法进行一定改进，使其更加合理，并给出对比结果 24](#_Toc108895962)

[8.可使用的参考文献 25](#_Toc108895963)

[数据算法文档结束！ 26](#_Toc108895964)

[前端文档 27](#_Toc108895965)

[**1.前端设计框架选取及理由** 27](#_Toc108895966)

[**2.界面原型设计** 27](#_Toc108895967)

[**2.1原型图** 27](#_Toc108895968)

[**2.2原型图设计原理** 28](#_Toc108895969)

[**3.系统功能设计** 29](#_Toc108895970)

[**3.1传递数据** 29](#_Toc108895971)

[**3.2呈现数据** 29](#_Toc108895972)

[**3.2.1用户管理数据呈现** 29](#_Toc108895973)

[**3.2.2预测信息与天气信息数据呈现** 29](#_Toc108895974)

[**4.工程内容（陈泽锋）** 30](#_Toc108895975)

[**4.1登录界面实现** 30](#_Toc108895976)

[**4.1.1登录界面展示** 31](#_Toc108895977)

[**4.1.2登录界面描述** 31](#_Toc108895978)

[**4.2注册界面实现** 31](#_Toc108895979)

[**4.2.1注册界面展示** 31](#_Toc108895980)

[**4.2.2注册界面描述** 32](#_Toc108895981)

[**4.3用户管理界面实现** 32](#_Toc108895982)

[**4.3.1用户管理界面展示** 32](#_Toc108895983)

[**4.3.2用户管理界面描述** 33](#_Toc108895984)

[**4.4界面美化** 34](#_Toc108895985)

[**5.工程内容（蒋涵）** 34](#_Toc108895986)

[**5.1主界面实现** 34](#_Toc108895987)

[**5.1.1主界面展示** 34](#_Toc108895988)

[**5.1.2主界面描述** 34](#_Toc108895989)

[**6.技术要点** 36](#_Toc108895990)

[**前端问题1：中国地图展示** 36](#_Toc108895991)

[**前端问题2：基于particles实现粒子背景，按钮无法点击触发** 37](#_Toc108895992)

[**前端问题3：界面优美排版问题** 37](#_Toc108895993)

[**前端问题4：界面美化问题** 38](#_Toc108895994)

[**前端问题5：机场设置** 39](#_Toc108895995)

[**前端问题6：vue+flask传值问题** 40](#_Toc108895996)

[前端文档结束！ 41](#_Toc108895997)

[后端文档 42](#_Toc108895998)

[1.后端框架选取及理由 42](#_Toc108895999)

[2.数据库对接工作 42](#_Toc108896000)

[2.1数据库配置对接 42](#_Toc108896001)

[2.2 SQL语句 43](#_Toc108896002)

[3. API编写 43](#_Toc108896003)

[3.1后端接口 43](#_Toc108896004)

[3.1.1利用路由进行传输 43](#_Toc108896005)

[3.1.2传输协议：HTTP协议 43](#_Toc108896006)

[3.2模块接口 44](#_Toc108896007)

[3.3登录接口login（）方法 44](#_Toc108896008)

[3.4注册接口signup（）方法 45](#_Toc108896009)

[3.5删除用户接口deleteUser（）方法 45](#_Toc108896010)

[3.6判断权限接口judgeAdmin（）方法 45](#_Toc108896011)

[3.7人员查询接口selectAlluser（）方法 45](#_Toc108896012)

[3.8选择起始机场接口setDepartureAirport（）方法 46](#_Toc108896013)

[3.9 选择到达机场接口setArriveAirport（）方法 46](#_Toc108896014)

[3.10 延误预测接口delayPredict（）方法 46](#_Toc108896015)

[3.11获取出发天气接口getDeaptureWeather（）方法 46](#_Toc108896016)

[3.12获取到达天气接口getDeaptureWeather（）方法 47](#_Toc108896017)

[4.系统功能逻辑实现 47](#_Toc108896018)

[4.1登录功能 47](#_Toc108896019)

[4.2注册功能 47](#_Toc108896020)

[4.3删除用户功能 48](#_Toc108896021)

[4.4选择起始机场 48](#_Toc108896022)

[4.5选择到达机场 48](#_Toc108896023)

[4.6延误预测功能 49](#_Toc108896024)

[4.7获取出发机场天气 49](#_Toc108896025)

[4.8获取到达机场天气 49](#_Toc108896026)

[4.9判断权限功能 49](#_Toc108896027)

[4.10列出所有用户 50](#_Toc108896028)

[5.技术要点 50](#_Toc108896029)

[5.1要点1：捕捉前端请求 50](#_Toc108896030)

[5.2要点2：vue+flask跨域访问问题 50](#_Toc108896031)

[5.3要点3：vue+flask传值问题 51](#_Toc108896032)

[6.API测试 52](#_Toc108896033)

[后端文档结束！ 52](#_Toc108896034)

[结语： 53](#_Toc108896035)

[附： 53](#_Toc108896036)

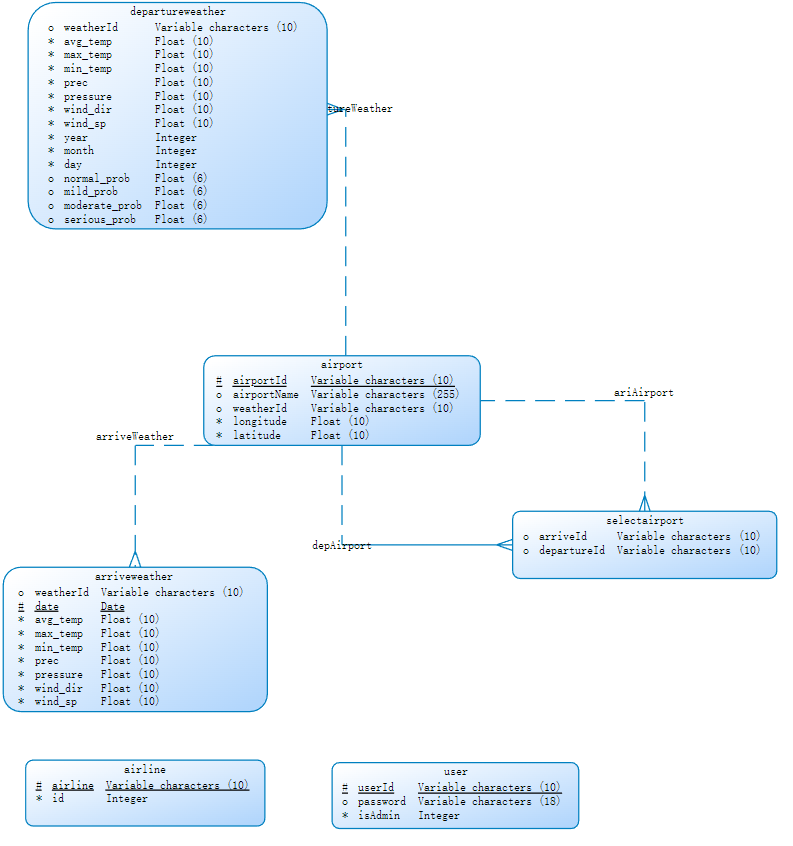
数据/算法文档

# 1.数据库对接、搭建工作

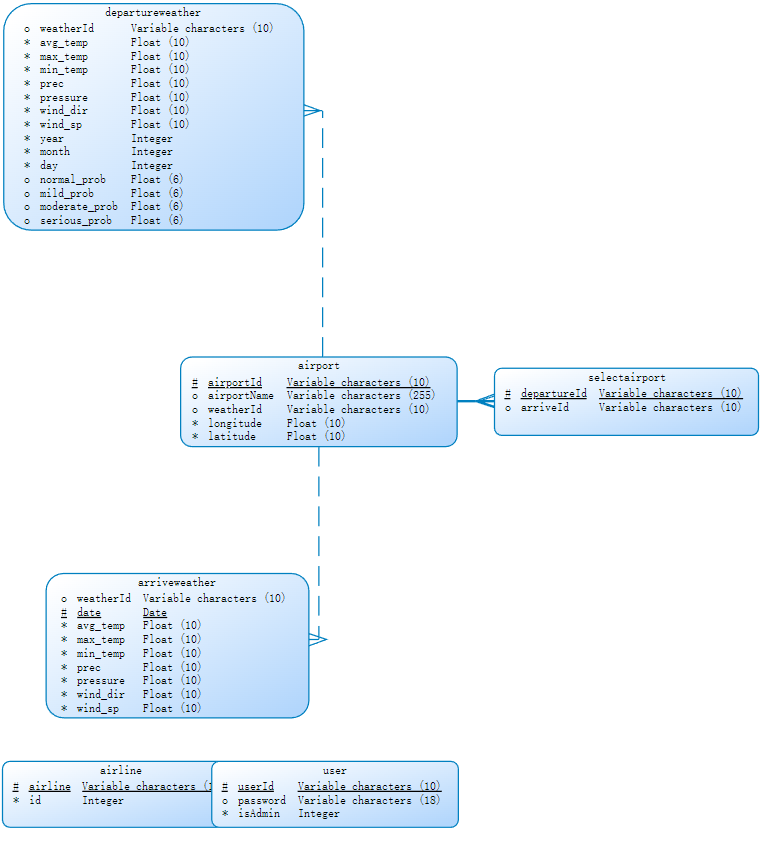
## 1.1搭建合适的数据库

使用mysql数据库，powerdesigner进行概念模型与物理模型的初步搭建

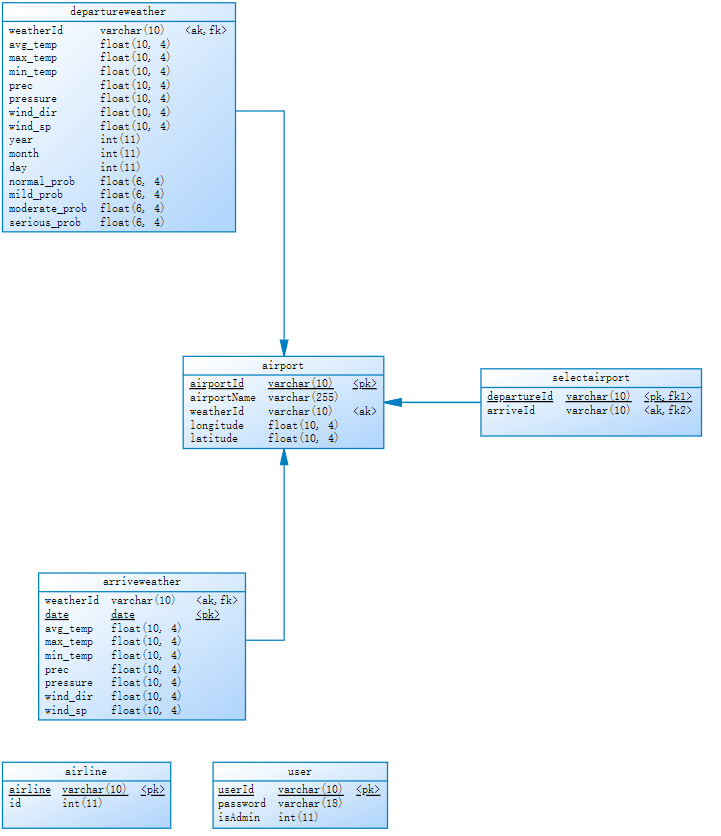
### 1.1.1概念模型



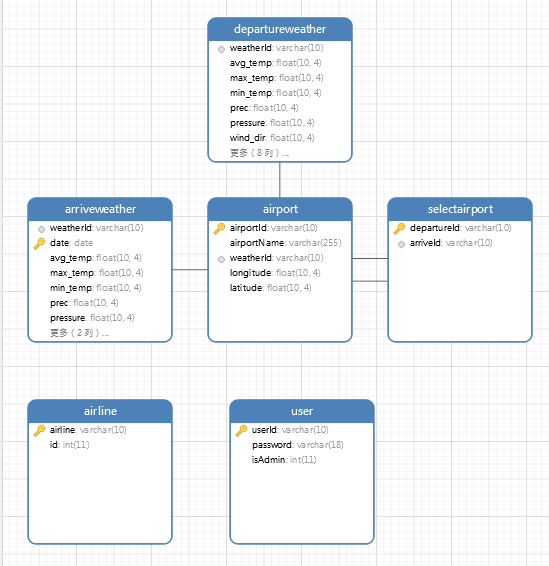
### 1.1.2逻辑模型



### 1.1.3物理模型

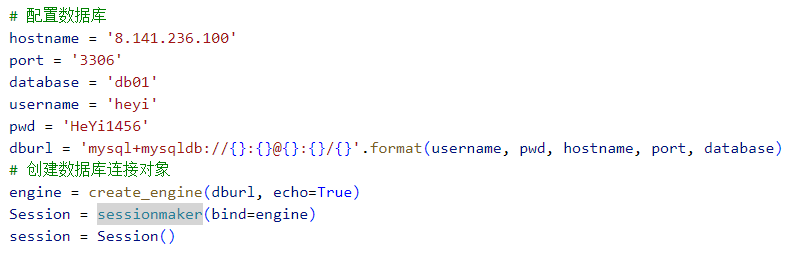


### 1.1.4ER图



## 1.2配合后端完成数据库的对接工作

### 1.2.1数据库配置对接

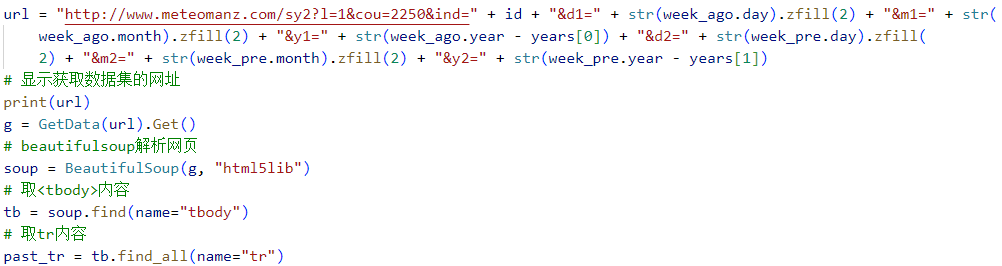


使用session便可以使用sql语句对数据库进行操作

# 2.气象预测算法寻找数据

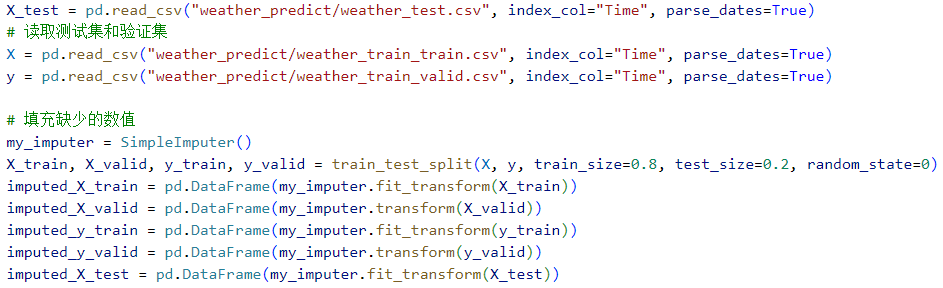
训练数据来源于[http://www.meteomanz.com/](http://www.meteomanz.com/" \t "dlt)

通过BeautifulSoup类库进行爬取



# 3.气象预测算法数据预处理

## 3.1计算均值填充缺少的数值

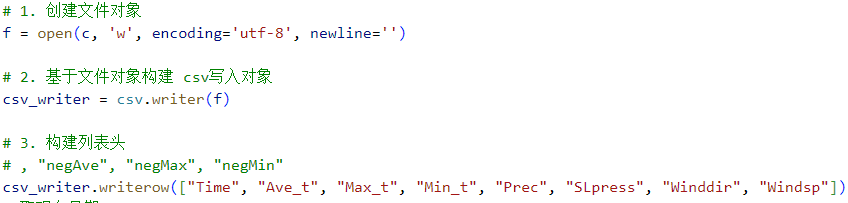


# 4.气象预测算法的实现

## 4.1使用GetData.py编写相关的爬虫程序爬取数据



## 4.2通过Write.py进行爬取数据的分析、清洗；之后写到创建好的csv文件中进行下一步的数据预处理





## 4.3通过ProcessData.py调用Write中的方法write进行数据的读取与csv文件的写入，并通过这些文件进行数据集的处理与划分。



## 4.4最后模型构建在GetModel.py中，使用随机森林模型，填入处理好的数据集，进行模型的训练与保存

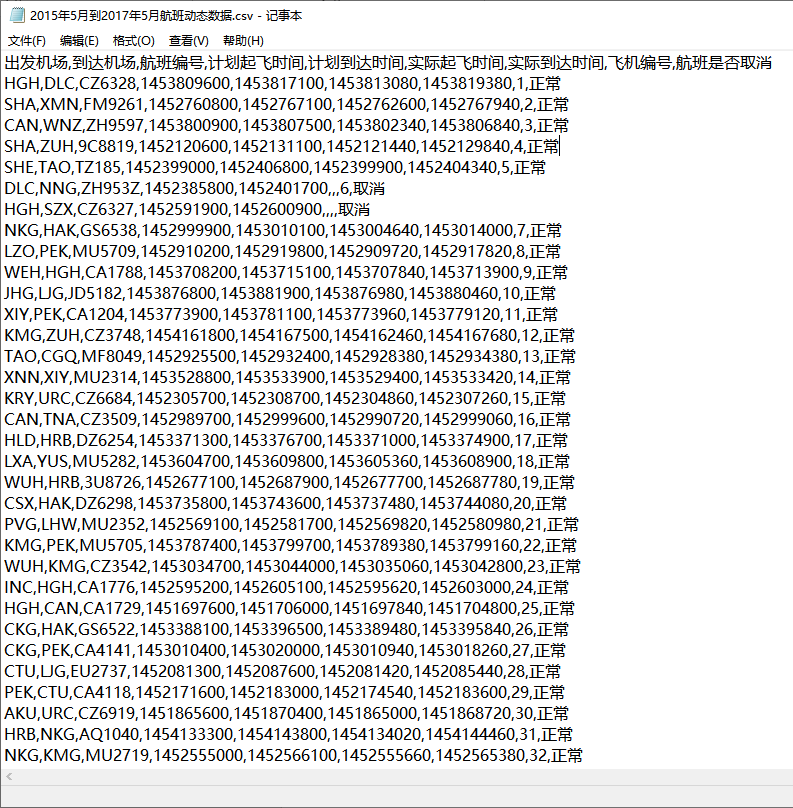


# 5.航班延迟预测算法寻找数据

## 5.1寻找到合适的数据集

航班动态起降数据集 - Heywhale.com

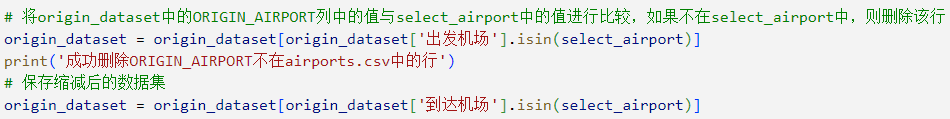
https://www.heywhale.com/mw/dataset/59793a5a0d84640e9b2fedd3



# 6.航班延迟预测算法数据预处理

使用jupyter notebook进行数据集处理

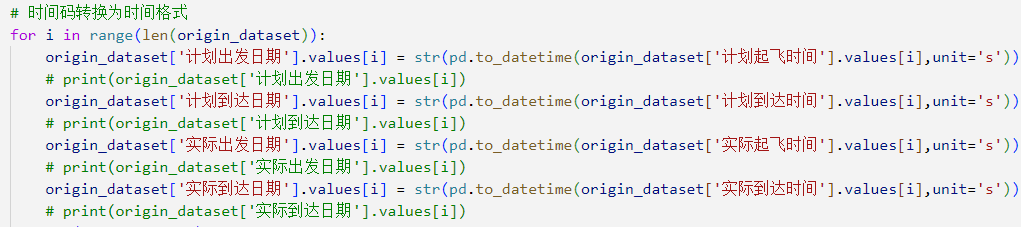
## 6.1缩减数据集，删除不是选中机场的出发机场和到达机场的行



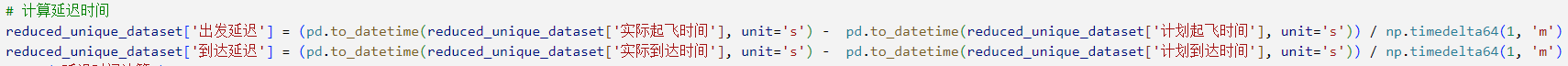
## 6.2删除取消的航班



## 6.3将时间戳转化为普通时间

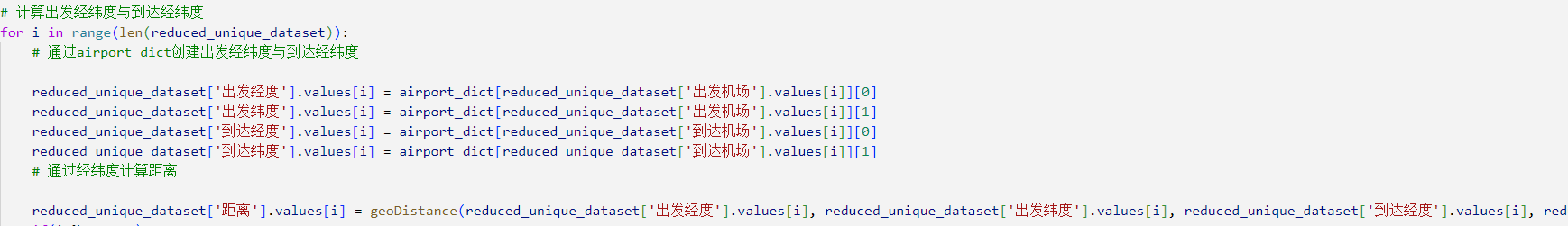


## 6.4延迟时间计算

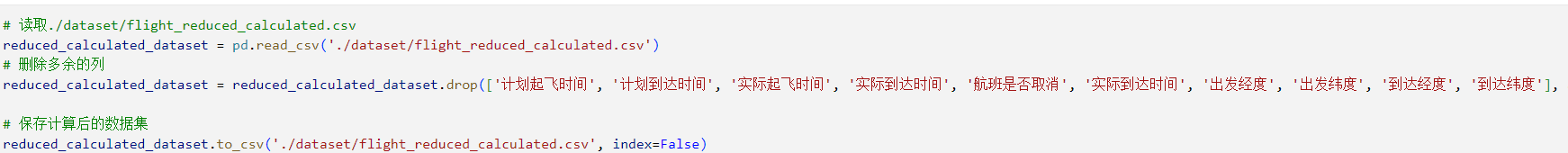


## 6.5通过两地经纬度计算距离

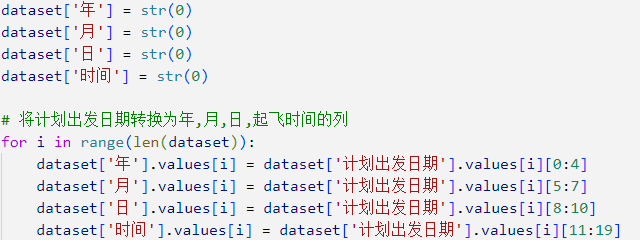




## 6.6删除多余的列



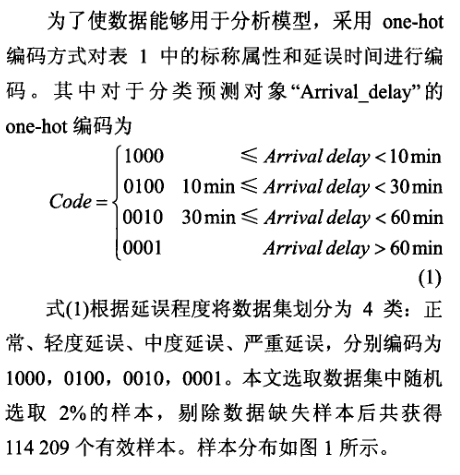
## 6.7将计划出发日期转化为年，月，日，起飞时间的列



## 6.8将出发延迟转化为分类预测对象

参考[1]唐红, 王栋, 宋博,等. 基于非线性赋权XGBoost算法的航班延误分类预测[J]. 系统仿真学报, 2021.

将延迟时间转化为延迟程度





## 6.9爬取所有地点的2015-2017年每天的气象数据



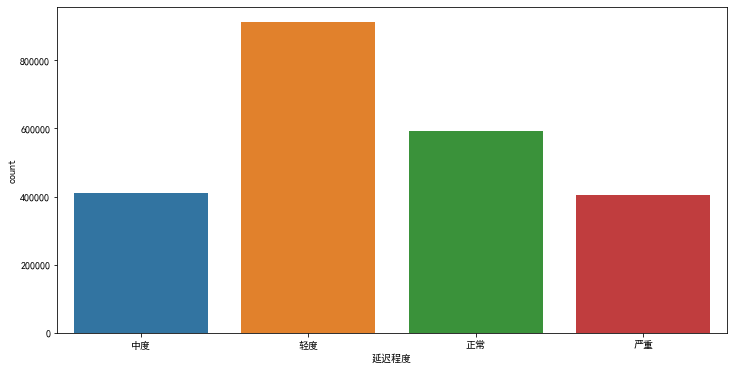
## 6.10将气象数据整合到处理好的航班延误数据集中，合并为最终的完整数据集



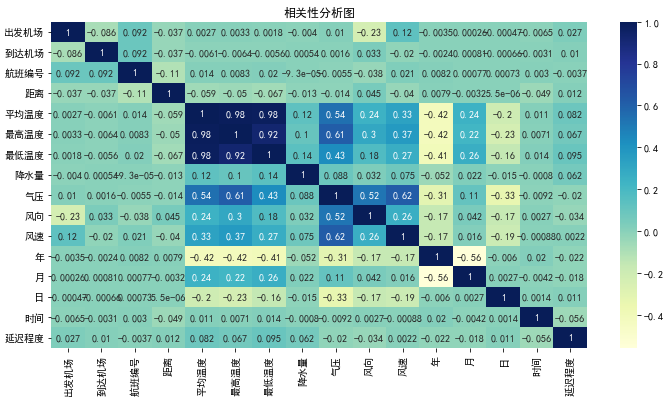
# 7.航班延迟预测算法的实现

## 7.1数据分析

### 7.1.1延迟程度频次统计图



### 7.1.2相关性分析

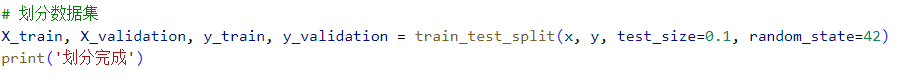


## 7.2使用基本的预测算法，成功实现基础预测功能

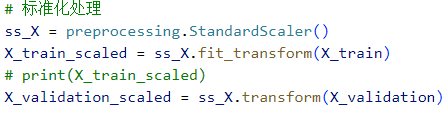
### 7.2.1将字符型数据转化为字典映射编码



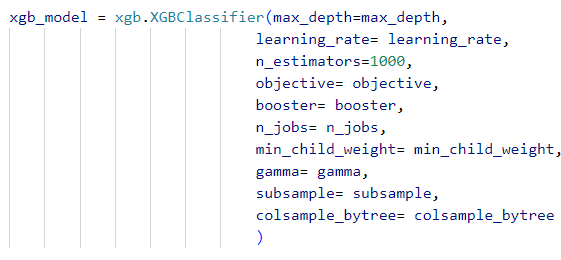
### 7.2.2划分数据集



### 7.2.3标准化处理

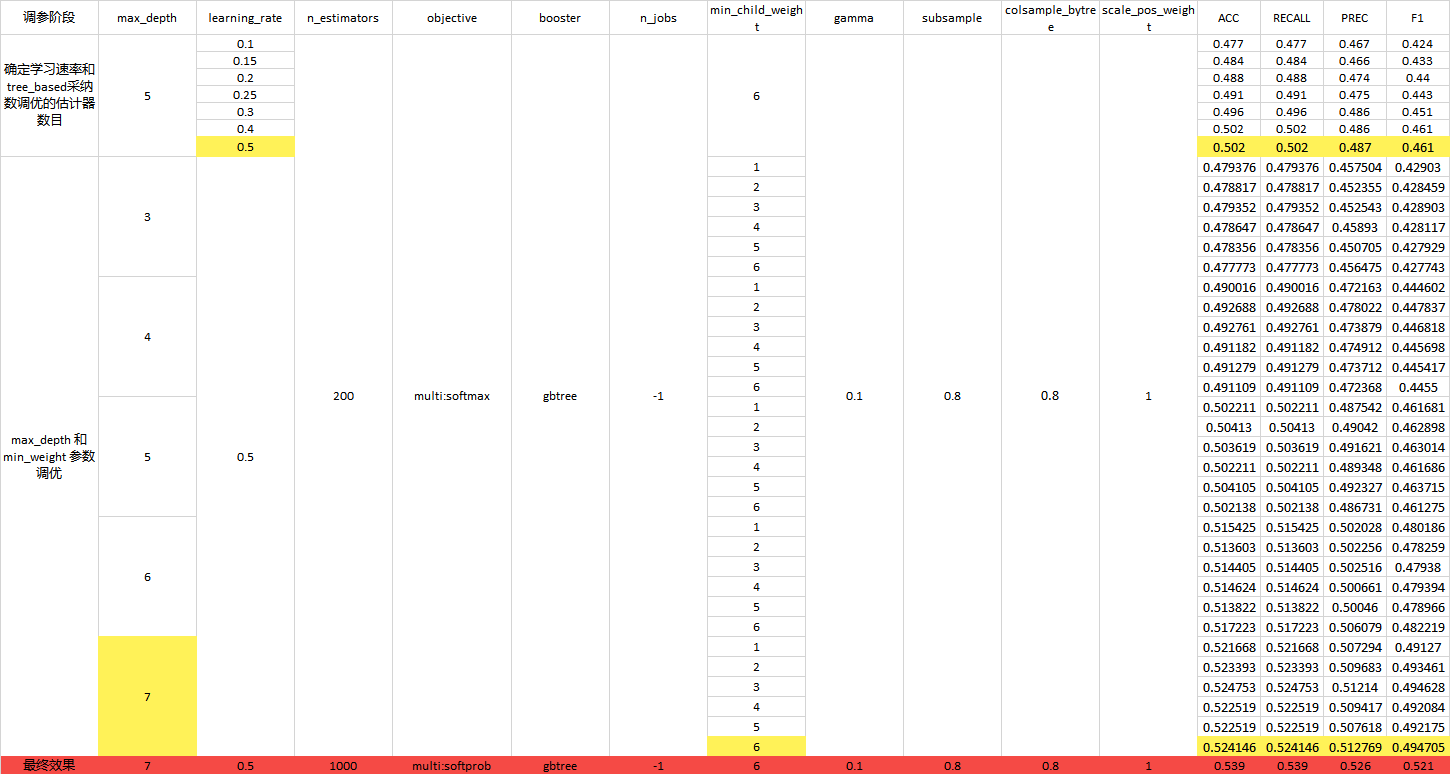


### 7.2.4建模

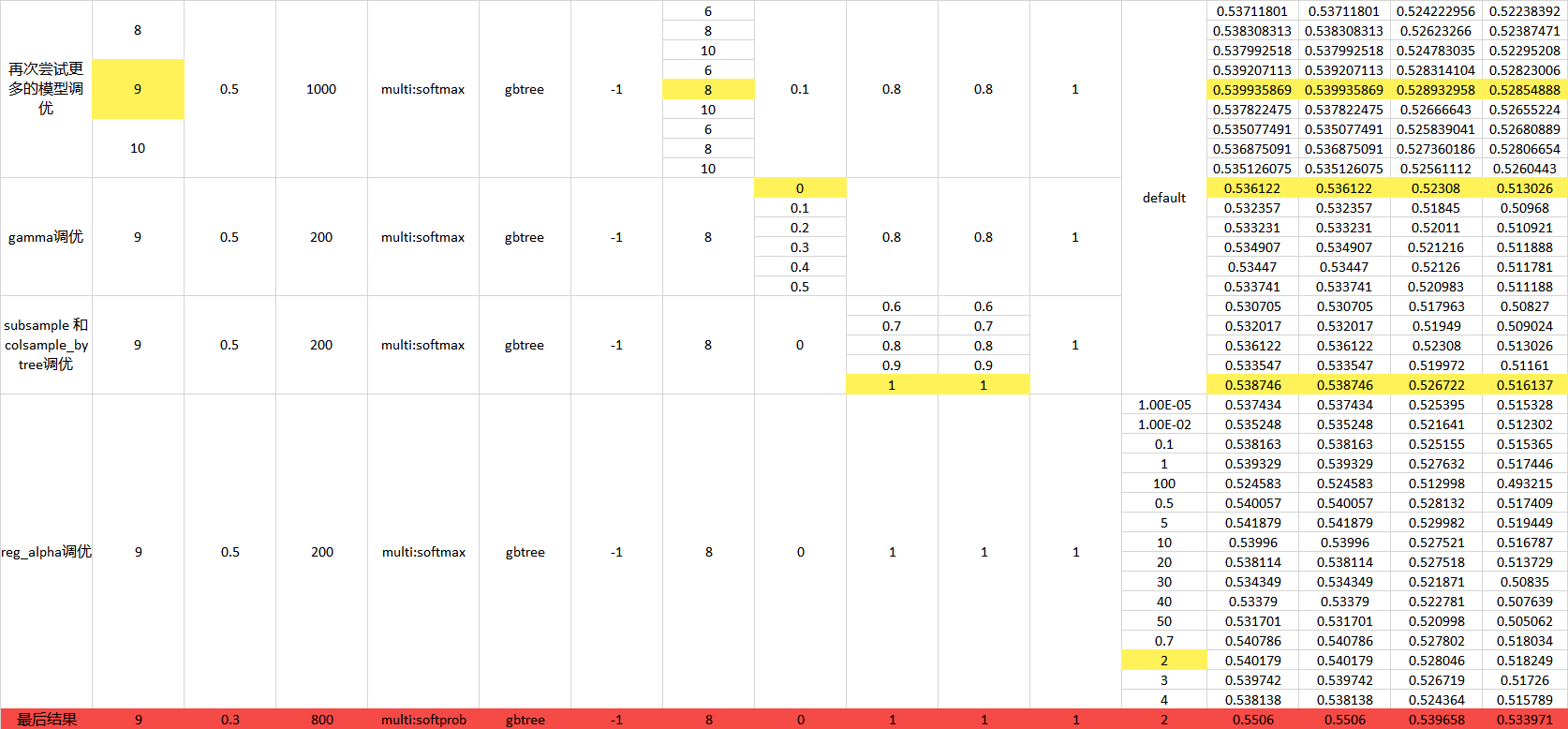


## 7.3对算法进行一定改进，使其更加合理，并给出对比结果

飞书表格链接地址：<https://lw547edakg.feishu.cn/sheets/shtcn46F5THkWBeZmcSjfCb7zLU>



首先对模型进行两个阶段的调参，第一阶段7次实验确定了学习率为0.5，第二阶段30次确定了最大深度为7，最小叶子节点权值为6。



然后继续又继续尝试了gamma, subsample, colsample\_bytree以及reg\_alpha的调优，将acc和f1尽可能地优化，通过75次的模型测试，花费近20个小时，最后达到acc为55.06%, f1score为53.397%。

与最开始的acc为47.7%，f1score为42.4%有着较为显著的提升。

# 8.可使用的参考文献

[1]中国民用航空局．2019年全国民航工作会议专题速递[EB/OL][2020-0616].http://www.caac.gov.cn/ZTZL/RDZT/2019Q GMHGZHY/.

[2]Chen T Q, Guestrin C. XGBoost: a scalable tree boosting system[C]//ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Francisco:USA: ACM, 2016: 785-794

[3]Chen T Q, He T, Benesty M. Xgboost: Extreme Gradient Boosting[R], R Package Version 0.4-2, 2015: 1

[4]刘金元，丁勇，李涛．基于梯度提升决策树的航班延误分类预测[J]．数学的实践与认识，2018，48(4): 1-7.

[5]唐红, 王栋, 宋博,等. 基于非线性赋权XGBoost算法的航班延误分类预测[J]. 系统仿真学报, 2021.

[6]董念清. 中国航班延误的现状、原因及治理路径[J]. 北京航空航天大学学报(社会科学版), 2013, 26(6):25-32.

[7]唐国利, 任国玉. 近百年中国地表气温变化趋势的再分析[J]. 气候与环境研究, 2005, 10(4):8.

# A数据算法文档结束！

前端文档

**1.前端设计框架选取及理由**

**两人商讨决定前端以vue+elementUI来实现，vue框架有着它自己的优势，首先它作为轻量级框架已经能够很好的满足我们该项目前端窗口的渲染实现，同时它简单易学，而elementUI则是有许多的可以直接使用的组件，在前端渲染的过程中有很好的适用性，同时也简单易懂。同时elementUI有以下的优点：**

**1、一致性**

**与现实生活一致：与现实生活的流程、逻辑保持一致，遵循用户习惯的语言和概念；**

**2、反馈**

**控制反馈：通过界面样式和交互动效让用户可以清晰的感知自己的操作；**

**3、效率**

**简化流程：设计简洁直观的操作流程；**

**清晰明确：语言表达清晰且表意明确，让用户快速理解进而作出决策；**

**4、可控**

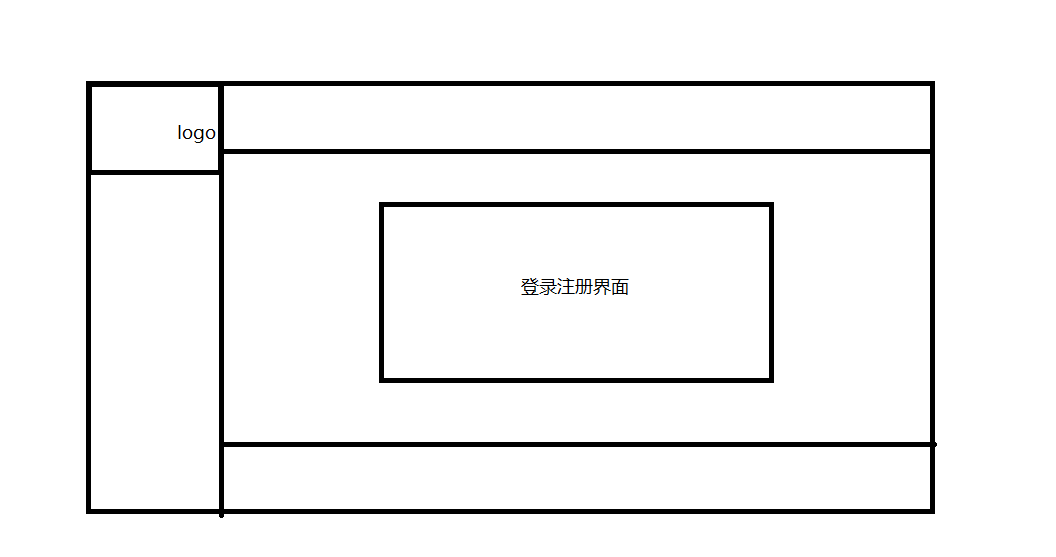
**用户决策：根据场景可给予用户操作建议或安全提示，但不能代替用户进行决策；**

**结果可控：用户可以自由的进行操作，包括撤销、回退和终止当前操作等。**

**2.界面原型设计**

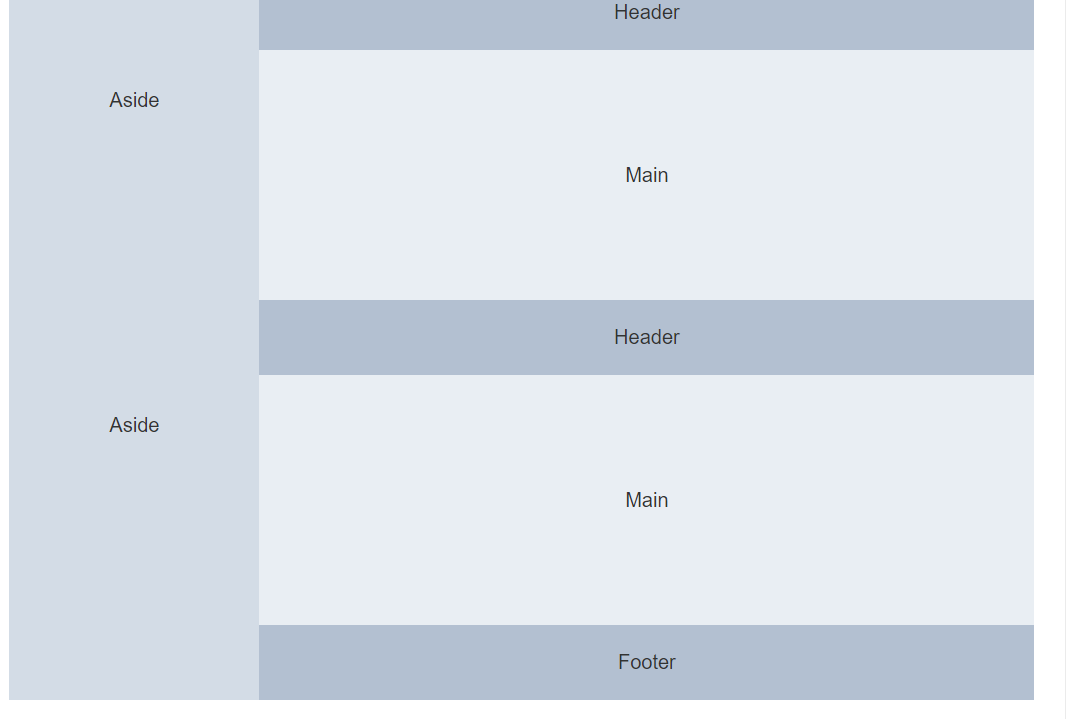
**2.1原型图**

**以该布局为主体，只对中间区域进行修改 其他区域保持一致，在左边的布局则实现界面切换的其他按钮，以及其他一些功能按钮。**



**2.2原型图设计原理**

**该原型图设计原理是源于vue+elementUI，在elementUI里有Container的布局容器，而其中就有该布局，同时美观好看，符合我们前端的设计原理，并且易于实现。同时登录注册界面位于main布局中，符合人性化交互设计，整个界面简洁大方，功能齐全，同时简化了流程，提高了用户的操作效率。**



**3.系统功能设计**

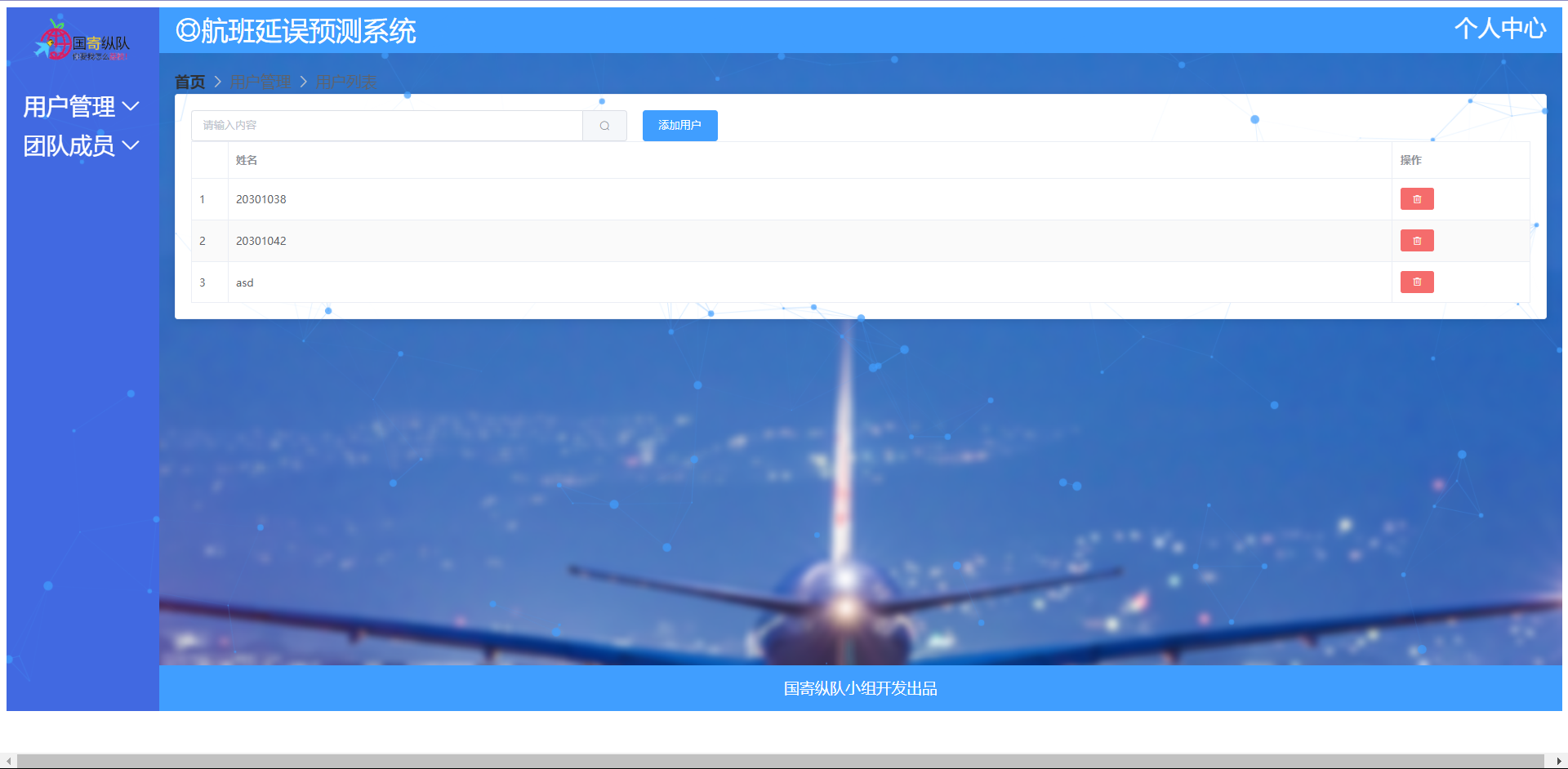
**3.1传递数据**

**前端在传递数据的时候采用axios，axios的优点在于从node.js创建http请求，在浏览器中创建XMLHttpRequest，同时支持Promise API，并且提供了一些并发请求的接口，支持拦截请求和响应，能够转换请求和响应数据，自动转换JSON数据，传值的时候以json传值。而传值，我们采取的是axios中的post方法来进行传值，将定义好的Form表单以json数据传输到后端。**

**3.2呈现数据**

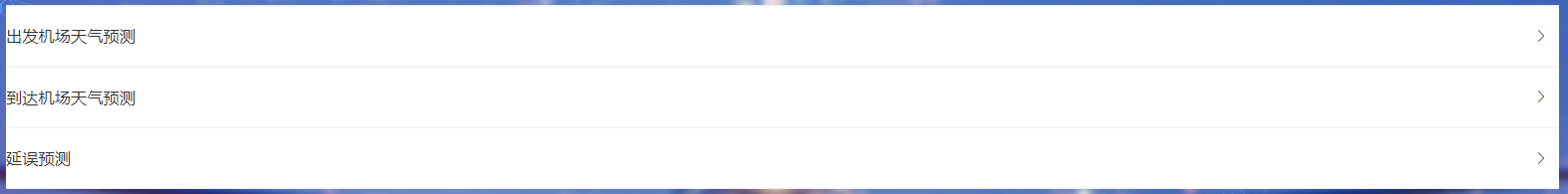
**3.2.1用户管理数据呈现**

**前端用axios，用get方法传递请求，后端接收后返回值，前端接收以后打印在表格里。**



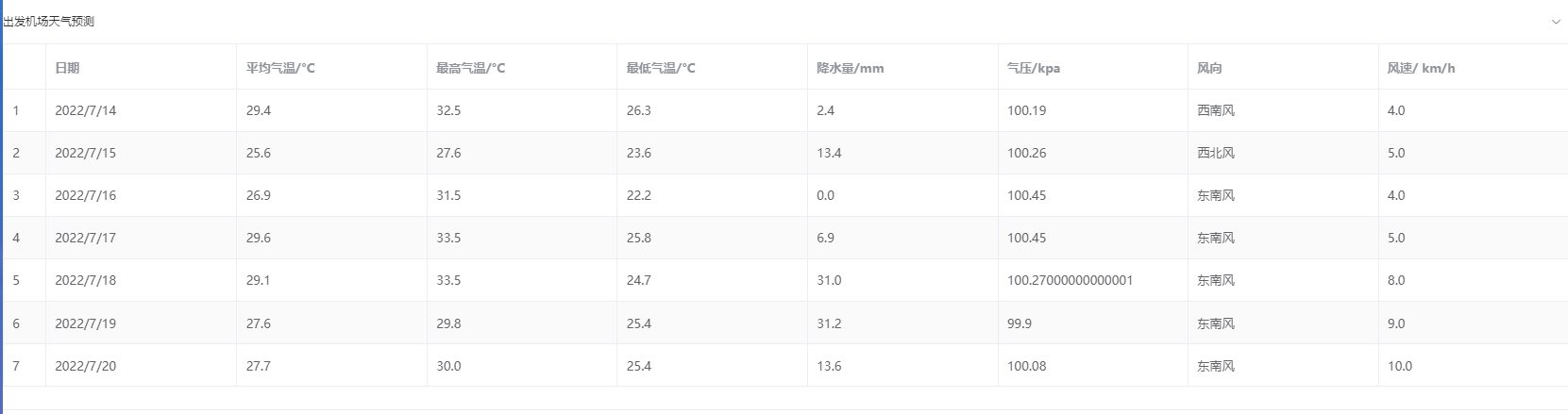
**3.2.2预测信息与天气信息数据呈现**

**使用折叠面板**



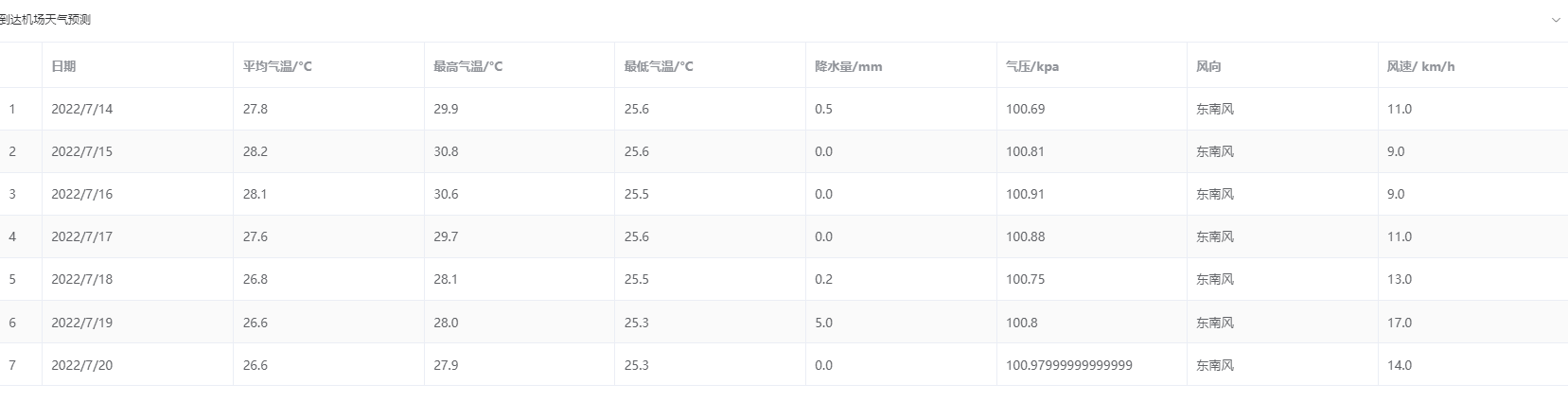
**出发天气信息**

**用户选择出发机场的同时就进行出发天气的预测**



**目的地天气信息**

**用户选择目的机场的同时就进行出发天气的预测**



**延误预测信息**

**用户选择出发机场，目的机场，时间后点击确定就开始进行延误信息预测**

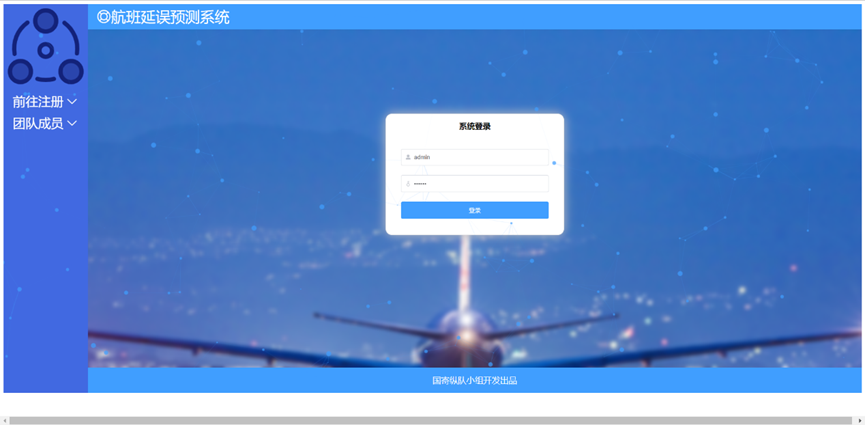


**4.工程内容（陈泽锋）**

**4.1登录界面实现**

**4.1.1登录界面展示**

**实现了登录界面的实现，调整窗口大小尺寸完成登录界面的窗口。**



**4.1.2登录界面描述**

**描述：提供用户登录功能**

**1.在文本框中输入用户名、密码**

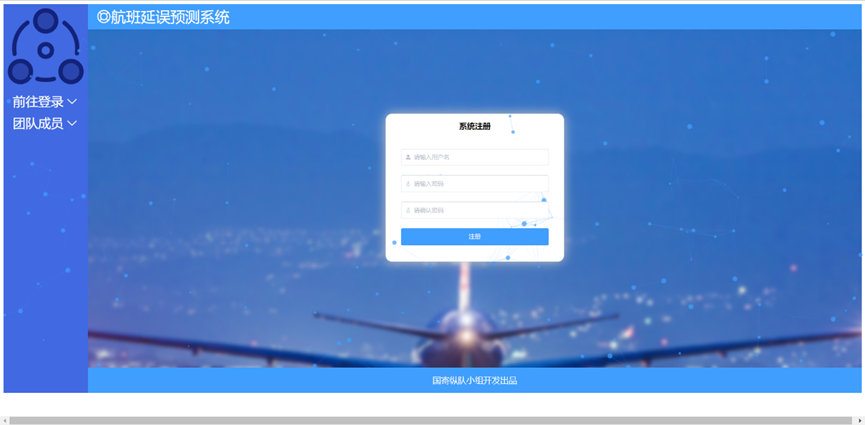
**2.点击登录按钮进行登录**

**3.登录成功后跳转到主界面，登录失败弹出提示框**

**4.2注册界面实现**

**4.2.1注册界面展示**

**实现了注册窗口的实现，调整窗口大小尺寸完成注册窗口的界面**



**4.2.2注册界面描述**

**描述：提供用户注册功能**

**1.在文本框中输入用户名、密码和确认密码**

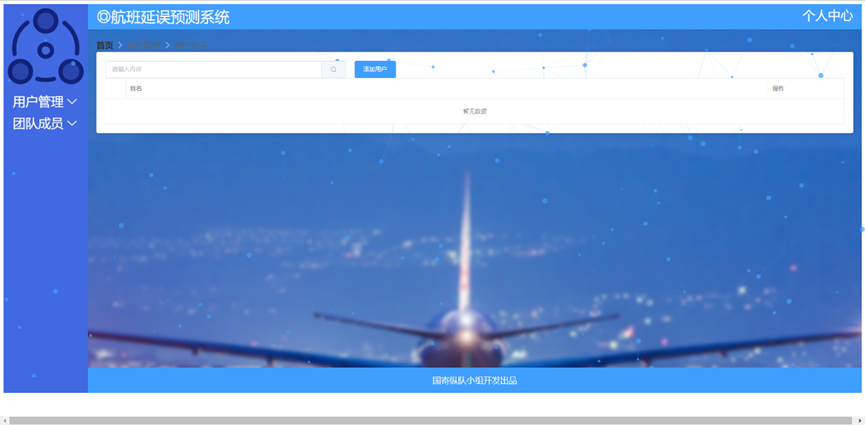
**2.点击注册按钮进行注册**

**3.注册成功后会跳转到登录界面，失败会弹出注册失败**

**4.3用户管理界面实现**

**4.3.1用户管理界面展示**

**实现了用户管理窗口的实现，调整窗口大小尺寸完成用户管理窗口的界面**

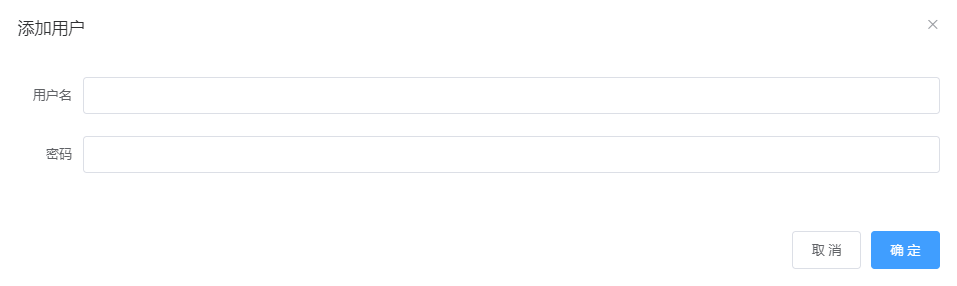


**4.3.2用户管理界面描述**

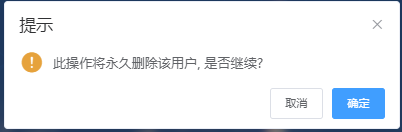
**描述：为管理员提供用户管理功能**

**1.可以输入用户的用户名，点击搜索按钮进行查询**

**2.可以点击添加用户按钮，在弹出框内输入用户名和密码进行添加用户，点击确定按钮，添加新用户，点击取消会取消添加新用户**



**3.可以在用户名后面点击删除按钮，进行删除用户，删除用户会进行确认，点击确定会删除用户，点击取消会取消删除用户的操作**



**4.4界面美化**

**为了使得前端渲染更加美观，同时吸引客户使用，前端在有背景图片的情况下，基于particles实现了粒子化背景，在粒子化背景中，会随着用户的鼠标生成一些可动的粒子跟随的效果，美化了该系统界面，使得界面更加好看。**

**5.工程内容（蒋涵）**

**5.1主界面实现**

**5.1.1主界面展示**

**实现了主界面窗口的实现，调整窗口大小尺寸完成主窗口的界面**



**5.1.2主界面描述**

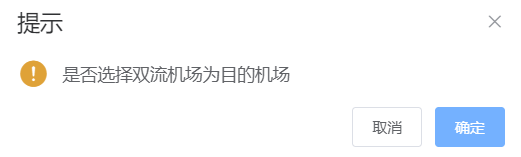
**描述：天气预测与延误预测**

**1.提示选择出发机场，单击机场按钮会弹窗**



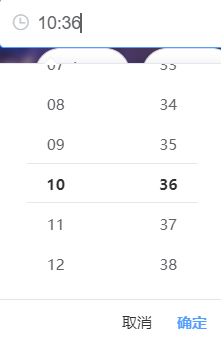
选择确定即可选择出发机场，取消则退出重选

**2.提示选择目的机场，单机机场按钮会弹窗**



选择确定即可选择目的机场，取消则退出重选

**3.提示选择时间，鼠标滚动即可调整时间**



点击确定即可选择时间

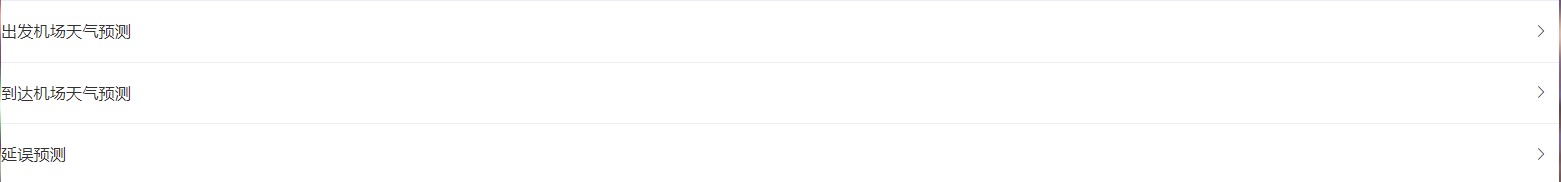
**4.确认按钮和取消按钮**



确认按钮点击即可生成天气预测信息和飞机延误预测信息

取消按钮则重置所有选择

**5.下拉框信息显示**



点击对应的下拉按钮即可查看预测信息

**6.技术要点**

**前端问题1：中国地图展示**

**解决：通过echarts完成中国地图的展示，但必须是4.9版本，因为新版本下删除了自带的地图**

**1、一些参数的简单说明**

**china: "echarts/map/json/china.json" 展示中国地图的包**

**tooltip：定制信息提示框的内容，其中params参数表示数据**

**visualMap：图注样式定制，其中包括color范围，文字提示**

**geo:定义地图为china，其中可能大概率要用的一个配置zoom: 视角缩放比例，roam：是否开启缩放和平移**

**itemStyle：地图外观定制，其中normal表示正常显示的样式，emphasis表示鼠标悬浮下样式**

**series：整体配置 其中type很关键 表示该例是地图，data:图表所用数据，实际项目中大家可以通过http获取数据，再赋值给data**



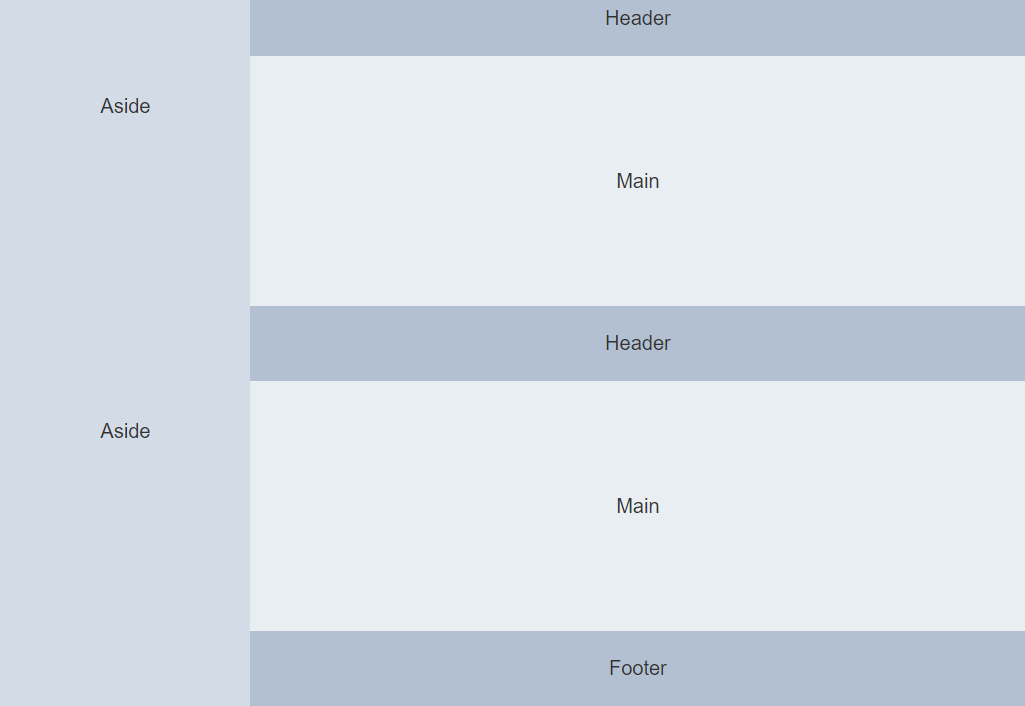
**前端问题2：基于particles实现粒子背景，按钮无法点击触发**

**解决：在按钮的内部加上position：relative；**

**相对定位，对象不可层叠，但将依据left，right，top，bottom等属性在正常文档流中偏移位置，在基于particles粒子背景的条件下，大部分按钮都无法点击，我认为这可能是图层的问题，在网上搜索以后，要在每个需要点击触发事件的组件里的css部分加上position：relative；才能让组件得以被点击触发。**

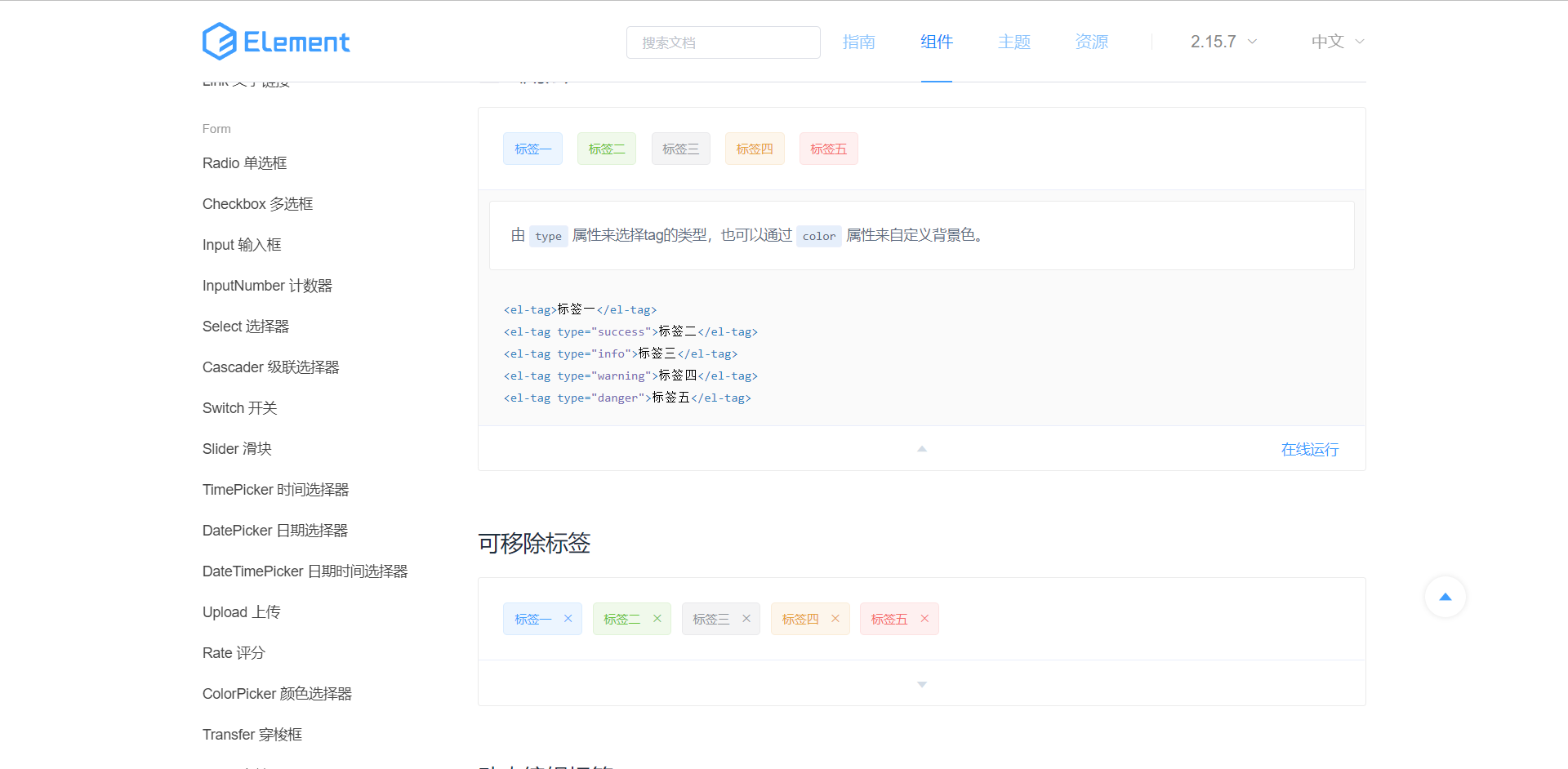
**前端问题3：界面优美排版问题**

**解决：通过<el-container>中分为aside，head，foot，main四个区域来实现，通过使用elementUI中的<el-container>中的<el-aside>，<el-head>和<el-foot>来进行布局，由此得到一个优美的界面排版，由此解决了前端界面的排版问题。**



**前端问题4：界面美化问题**

**解决：用elementui组件实现优美组件，element 生态更好，使用频率远超过iview ,element开发团队实力，同时，支持 Vue 2.x 组件库里最好的了，常用业务组件全面，功能丰富，有英文文档，生态齐全，支持 SSR。在elementUI里的按钮组件，输入框组件各种都适用于vue2的界面开发。**



**前端问题5：机场设置**

**解决：使用echart里的effectScatter散点图进行绘制**

**effectScatter：散点图，用于显示机场，实现机场点击事件**

**设置hoverAnimation为true即鼠标放在机场上散点图该点变粗**

**使用on click事件对地图的点击事件进行甄别，如果是effectScatter则判定为机场选择**

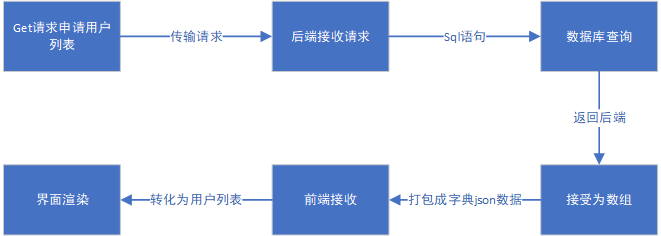


**前端问题6：vue+flask传值问题**

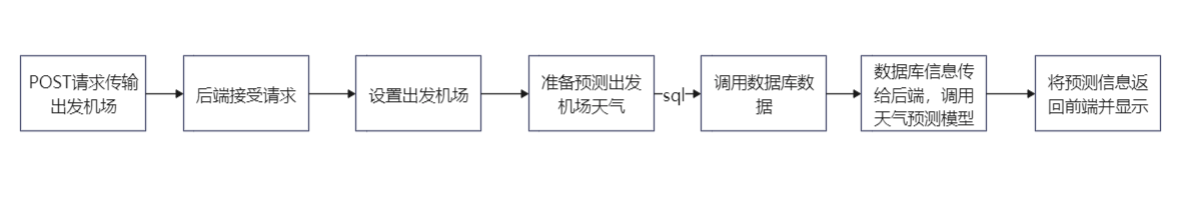
**解决：用post进行传值时，一开始是post传值，但我们并没有考虑到这个问题，在flask接收端没有设置post接收，因此一开始出现了网络问题，后来换成post接收以后，在flask返回值的地方出现了问题，前端不能直接接收，后来将传回来的值打包成json数据，借此，前端vue框架在接收json数据以后能够实现数据的传值和渲染，而对登录账户的权限判断由前端向后端传递登录的用户名，而后端进入数据库判断是否具有权限后返回true或者false的值，返回前端决定能否进行界面跳转。**



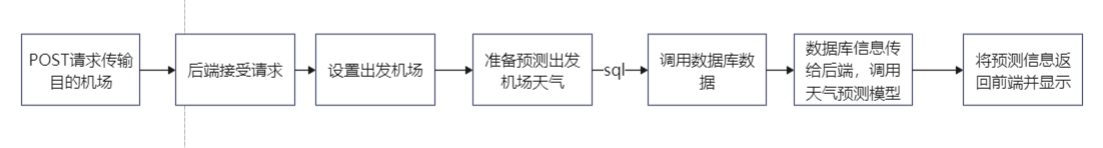
**用户管理权限判断流程图**



**用户列表获取渲染流程图**



**预测出发机场天气流程图**



**预测出发机场天气流程图**



预测延误信息流程图

# 前端文档结束！

后端文档

# 1.后端框架选取及理由

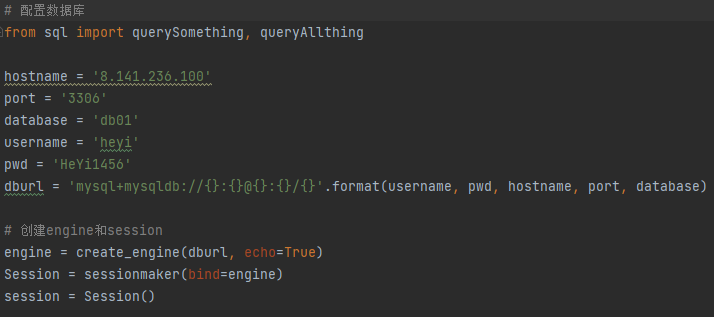
**后端框架选用的是Flask框架。[Flask](https://so.csdn.net/so/search?q=Flask&spm=1001.2101.3001.7020" \t "_blank)是一个Python编写的Web 微框架，让我们可以使用Python语言快速实现一个网站或Web服务。**

**优点:  
  -**[**框架**](https://so.csdn.net/so/search?q=%E6%A1%86%E6%9E%B6&spm=1001.2101.3001.7020)**很轻量  
  -  更新时依赖小  
  -  简洁  
  -  扩展性好  
  -**[**第三方库**](https://so.csdn.net/so/search?q=%E7%AC%AC%E4%B8%89%E6%96%B9%E5%BA%93&spm=1001.2101.3001.7020)**的选择面广，开发时可以结合自己喜欢用的轮子，也可以结合强大的python库**

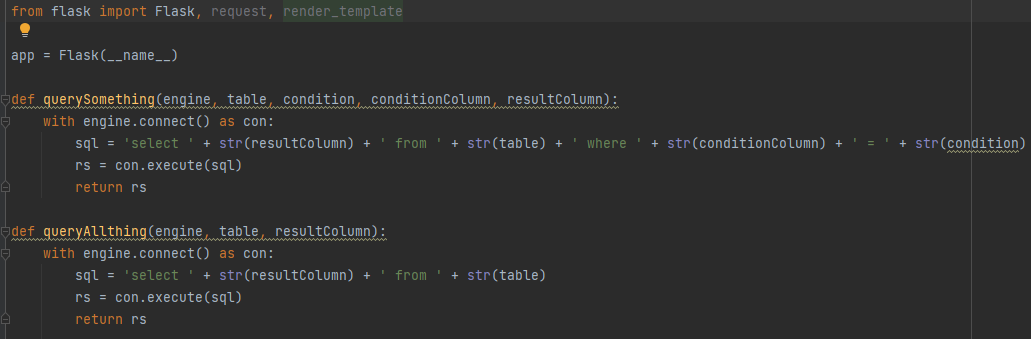
# 2.数据库对接工作

### 2.1数据库配置对接

使用session便可以使用sql语句对数据库进行操作



### 2.2 SQL语句



# API编写

## 3.1后端接口

### 3.1.1利用路由进行传输

主要分为以下几个步骤：

1. 浏览器发出请求
2. 服务器端听到端口有请求过来，并解析url路径
3. 根据服务器的路由设置，返回相应的信息（json数据等）
4. 浏览器根据数据包的种类决定怎样处理解析数据

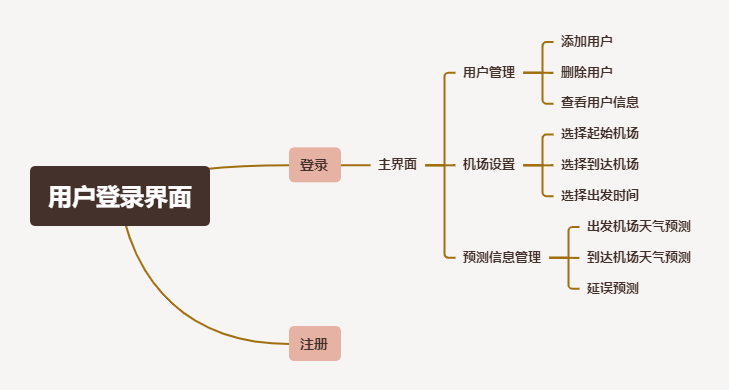
路由就是前端与后端服务器进行交互的一种方式，通过不同的路径来请求不同的资源，请求不同的页面是路由的一项功能。

### 3.1.2传输协议：HTTP协议

浏览器作为HTTP客户端通过URL向HTTP服务器即搭建的 flask框架的服务器发送所有请求。

本项目主要采用post请求方法。

## 3.2模块接口

本部分介绍模块间逻辑关系的接口设计，主要通过模块图来展现。

## 3.3登录接口login（）方法

前端发送用户的id和密码进行验证并返回登录结果

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 输入 | idNum | Int | 用户id |
| 输入 | password | String | 用户密码 |
| 返回 | success/fasle | String | 处理结果 |

## 3.4注册接口signup（）方法

前端发送新增用户的信息，系统进行处理并返回处理结果

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 输入 | id | Int | 用户id |
| 输入 | password | String | 用户密码 |
| 返回 | true | String | 处理结果 |

## 3.5删除用户接口deleteUser（）方法

前端发送删除用户的id，系统进行处理并返回处理结果

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 输入 | userid | Int | 用户id |
| 返回 | true | String | 处理结果 |

## 3.6判断权限接口judgeAdmin（）方法

前端发送用户id，系统判断其是否为管理员并返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 输入 | idNum | Int | 用户id |
| 返回 | true/false | String | 处理结果 |

## 3.7人员查询接口selectAlluser（）方法

前端发送查看请求，后端返回所有用户信息

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 返回 | allUser | List | 所有用户 |

## 3.8选择起始机场接口setDepartureAirport（）方法

前端发送出发机场名，系统进行出发机场天气预测并返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 输入 | departurAirport | String | 出发机场名称 |
| 返回 | pred | List | 出发机场天气预测信息 |

## 3.9 选择到达机场接口setArriveAirport（）方法

前端发送到达机场名，系统进行到达机场天气预测并返回结果

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 输入 | arriveAirport | String | 到达机场名称 |
| 返回 | pred | List | 到达机场天气预测信息 |

## 3.10 延误预测接口delayPredict（）方法

前端发送出发时间（小时），系统进行延误预测并返回预测结果

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 输入 | hour | Int | 出发时间 |
| 返回 | pred | List | 延误预测信息 |

## 3.11获取出发天气接口getDeaptureWeather（）方法

前端发送查看出发天气请求，系统返回出发天气

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 返回 | str(weatherList[0:len(weatherList)-1][1:11]) | List | 天气与延误信息二维列表 |

## 3.12获取到达天气接口getDeaptureWeather（）方法

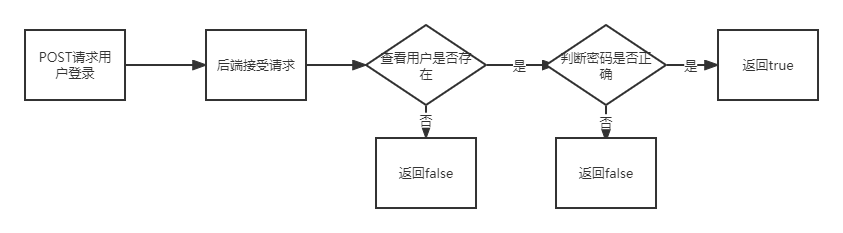
前端发送查看到达天气请求，系统返回到达天气

|  |  |  |  |
| --- | --- | --- | --- |
| 参数类别 | 名称 | 类型 | 描述 |
| 返回 | str(weatherList[0:len(weatherList)-1][1:11]) | List | 天气与延误信息二维列表 |

# 4.系统功能逻辑实现

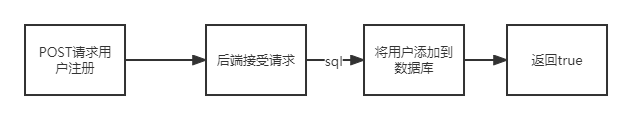
## 4.1登录功能

登录流程图



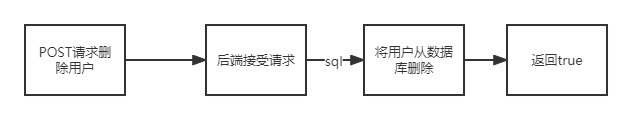
## 4.2注册功能

注册流程图



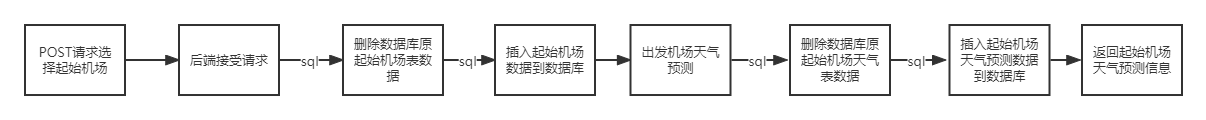
## 4.3删除用户功能

删除流程图



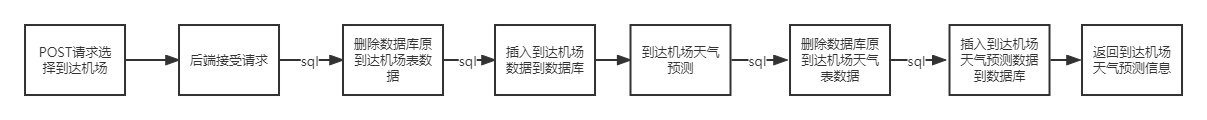
## 4.4选择起始机场

选择起始机场流程

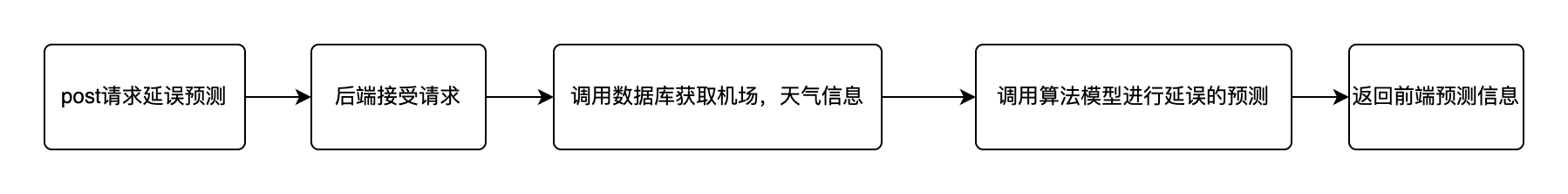


## 4.5选择到达机场

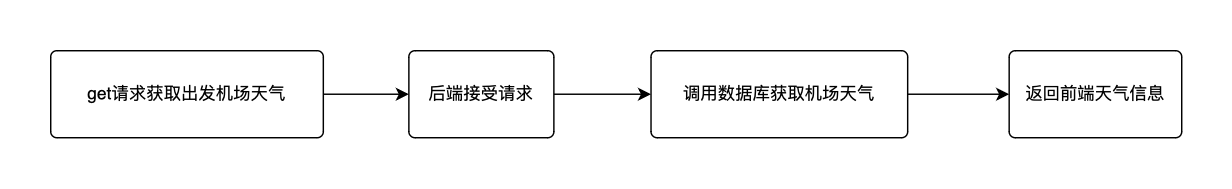
选择到达机场流程



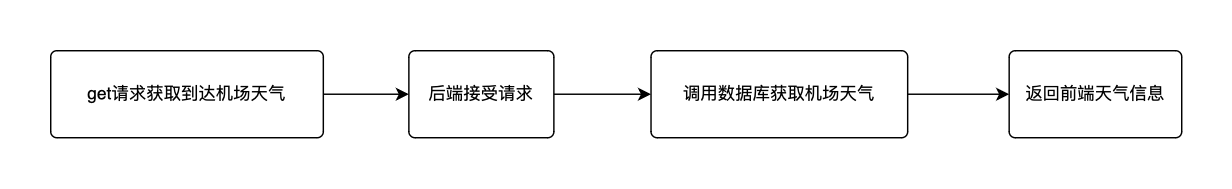
## 4.6延误预测功能



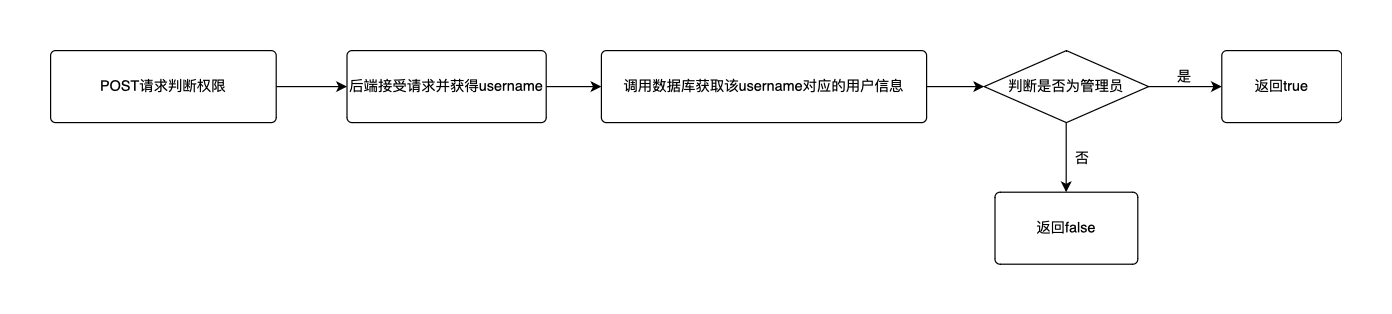
## 4.7获取出发机场天气



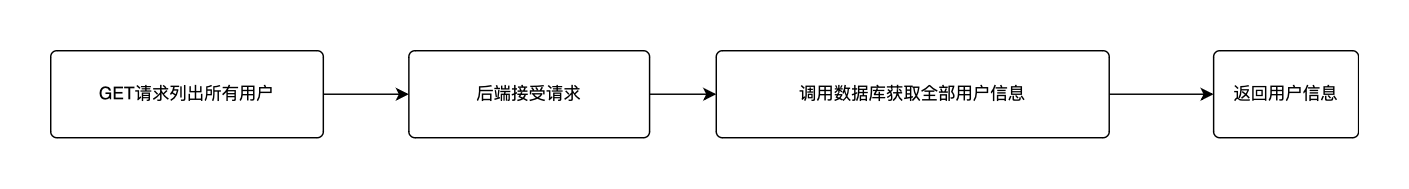
## 4.8获取到达机场天气



## 4.9判断权限功能



## 4.10列出所有用户



# 5.技术要点

## 5.1要点1：捕捉前端请求

默认情况下， Flask的路由支持HTTP的GET请求， 如果需要试图函数支持HTTP的其他方法， 可以通过methods关键字参数进行设置。 关键字参数methods的类型为list， 可以同时指定多种HTTP方法。

这两种请求的区别如下:

* GET把参数包含在URL中， 也就是直接输入网址访问， 把参数放到这个网址里面去的时候，访问的就是get请求。
* POST通过**request body**传递参数， 采用表单的时候往往就是这个。

## 5.2要点2：vue+flask跨域访问问题

**跨域**，指的是从一个域名去请求另外一个域名的资源。即跨域名请求。跨域时，浏览器不能执行其他域名网站的脚本，是由浏览器的同源策略造成的，是浏览器施加的安全限制。严格一点来说就是 只要协议，域名，端口有任何一个的不同，就被当作是跨域。

CORS，全称是"跨域资源共享"（Cross-origin resource sharing）。

它允许浏览器向跨源服务器，发出XMLHttpRequest请求，从而克服了AJAX只能同源使用的限制。

解决跨域问题可以使用两种方法：

1.使用@cross\_origin装饰器

2.使用CORS函数，应用全局配置

## 5.3要点3：vue+flask传值问题

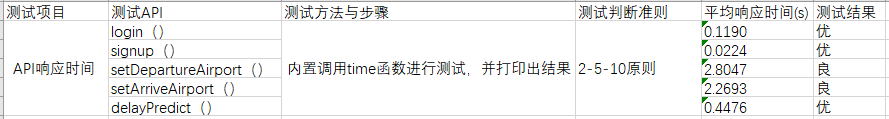
服务器在接收到客户端请求后，会自动创建request对象，request对象由Flask框架创建，可以给它添加属性，但不能修改或删除它的属性。

在本项目中，前端通过表单将值传输给后端，后段通过request.form("字段名")访问前端传回的数据。

后端往前端传输数据时，以flask Response对象进行传输。Response对象由开发人员构造，包含服务器返回给客户端的数据，Flask返回的Response包括这4种类型，string，response对象，tuple，WSGI instance实例。其中，我们一般会以json字符串以及tuple元组作为返回值传递到前端。

# 6.API测试

我们主要对API接口的响应时间进行了测试，通过内置调用time函数进行测试，并打印出结果。并根据2-5-10原则进行测试判断，可以看到最后各接口的测试结果还是比较好的。



# 后端文档结束！

结语：

感谢，

感谢鲁凌云老师的悉心安排与细心指导；

感谢助教张星宇学长对项目的理论性讲解与导入和对我们项目的帮助；

感谢小组成员每个人的努力；

对2022年夏大二下暑期实训学到了很多、内心感动了很多、对未来明确了很多致以最高的感谢！

附：

项目成员：

数据与算法工程师：贺思超

前端工程师：陈泽锋、蒋涵

后端工程师：何毅、江顺

项目经理：解世超

项目地址（GitHub）：

<https://github.com/Routhleck/flight-delay-predict>