

项目文档

软件学院

数据处理人员

作者：何毅，江顺



目录

[后端文档 5](#_Toc108787479)

[1.后端框架选取及理由 5](#_Toc108787480)

[2.数据库对接工作 5](#_Toc108787481)

[2.1数据库配置对接 5](#_Toc108787482)

[2.2 SQL语句 6](#_Toc108787483)

[3. API编写 6](#_Toc108787484)

[3.1后端接口 6](#_Toc108787485)

[3.1.1利用路由进行传输 6](#_Toc108787486)

[3.1.2传输协议：HTTP协议 6](#_Toc108787487)

[3.2模块接口 7](#_Toc108787488)

[3.3登录接口login（）方法 7](#_Toc108787489)

[3.4注册接口signup（）方法 8](#_Toc108787490)

[3.5删除用户接口deleteUser（）方法 8](#_Toc108787491)

[3.6判断权限接口judgeAdmin（）方法 8](#_Toc108787492)

[3.7人员查询接口selectAlluser（）方法 8](#_Toc108787493)

[3.8选择起始机场接口setDepartureAirport（）方法 9](#_Toc108787494)

[3.9 选择到达机场接口setArriveAirport（）方法 9](#_Toc108787495)

[3.10 延误预测接口delayPredict（）方法 9](#_Toc108787496)

[3.11获取出发天气接口getDeaptureWeather（）方法 9](#_Toc108787497)

[3.12获取到达天气接口getDeaptureWeather（）方法 10](#_Toc108787498)

[4.系统功能逻辑实现 10](#_Toc108787499)

[4.1登录功能 10](#_Toc108787500)

[4.2注册功能 10](#_Toc108787501)

[4.3删除用户功能 11](#_Toc108787502)

[4.4选择起始机场 11](#_Toc108787503)

[4.5选择到达机场 11](#_Toc108787504)

[4.6延误预测功能 12](#_Toc108787505)

[4.7获取出发机场天气 12](#_Toc108787506)

[4.8获取到达机场天气 12](#_Toc108787507)

[4.9判断权限功能 12](#_Toc108787508)

[4.10列出所有用户 13](#_Toc108787509)

[5.技术要点 13](#_Toc108787510)

[5.1要点1：捕捉前端请求 13](#_Toc108787511)

[5.2要点2：vue+flask跨域访问问题 13](#_Toc108787512)

[5.3要点3：vue+flask传值问题 14](#_Toc108787513)

[6.API测试 15](#_Toc108787514)

后端文档

# 1.后端框架选取及理由

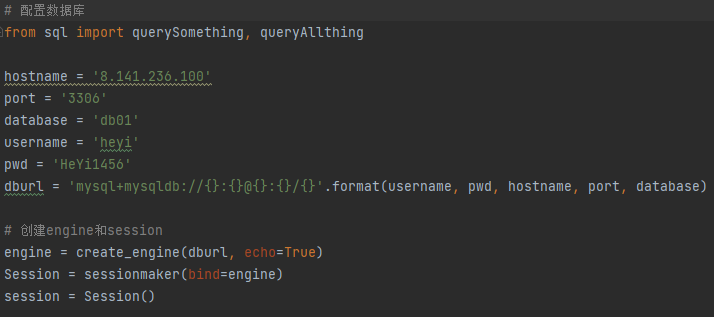
**后端框架选用的是Flask框架。[Flask](https://so.csdn.net/so/search?q=Flask&spm=1001.2101.3001.7020" \t "_blank)是一个Python编写的Web 微框架，让我们可以使用Python语言快速实现一个网站或Web服务。**

**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**

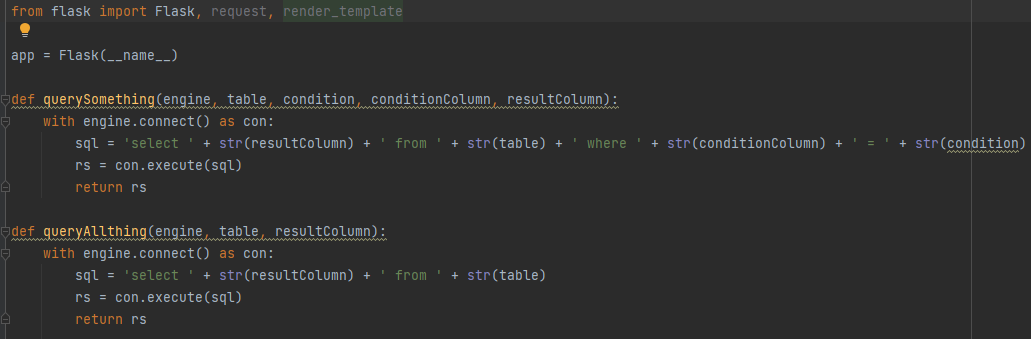
# 2.数据库对接工作

### 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

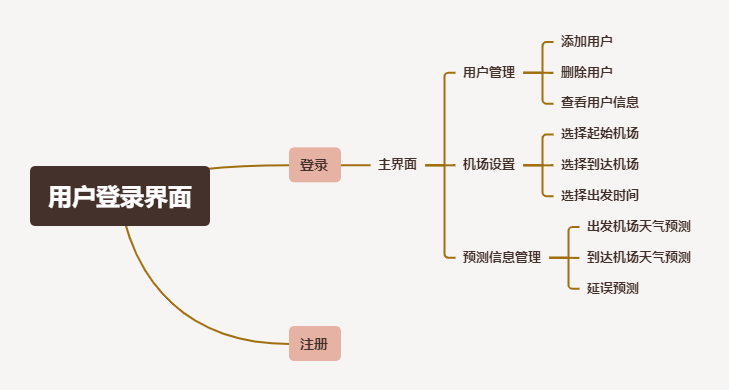
路由就是前端与后端服务器进行交互的一种方式，通过不同的路径来请求不同的资源，请求不同的页面是路由的一项功能。

### 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

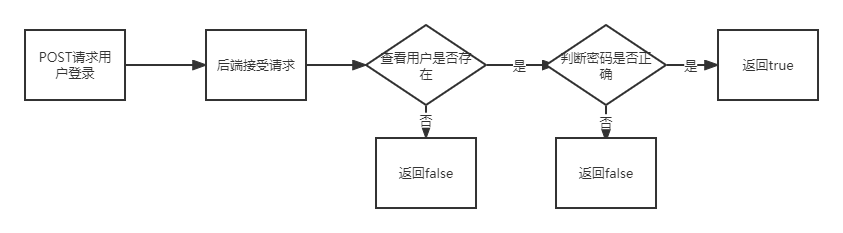
前端发送查看到达天气请求，系统返回到达天气

|  |  |  |  |
| --- | --- | --- | --- |
| 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

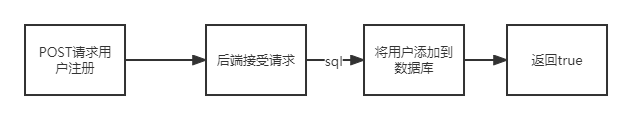
## 4.1登录功能

登录流程图



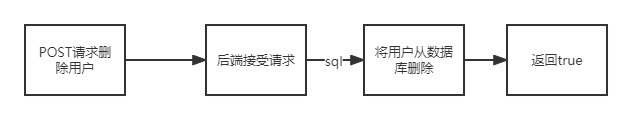
## 4.2注册功能

注册流程图



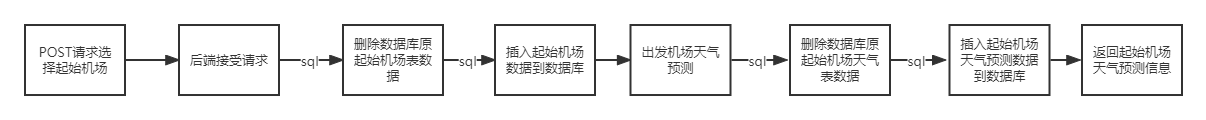
## 4.3删除用户功能

删除流程图



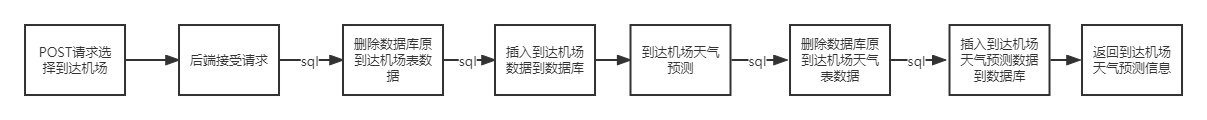
## 4.4选择起始机场

选择起始机场流程

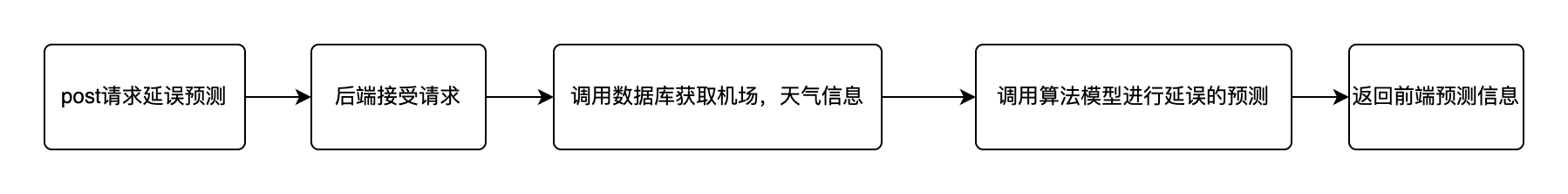


## 4.5选择到达机场

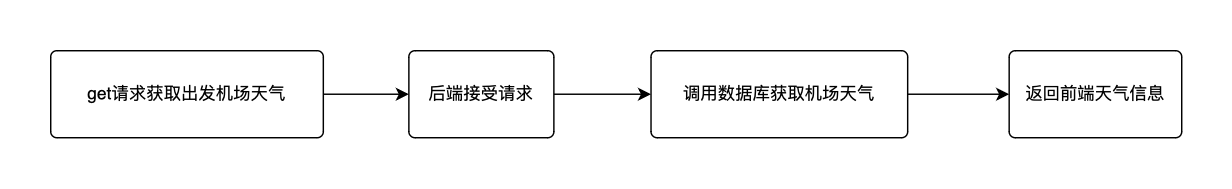
选择到达机场流程



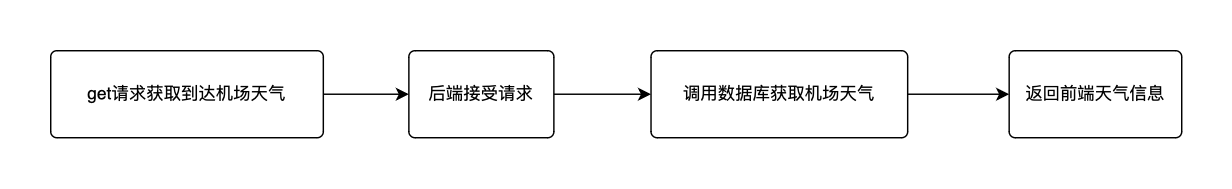
## 4.6延误预测功能



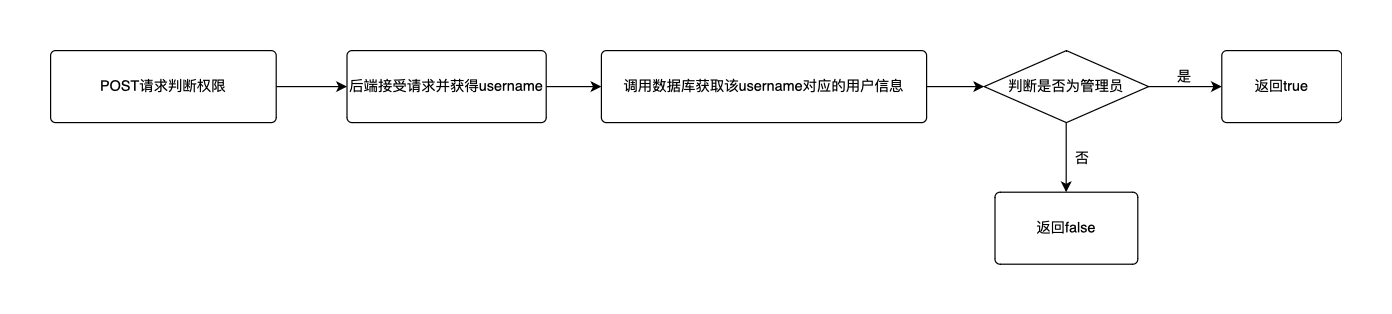
## 4.7获取出发机场天气



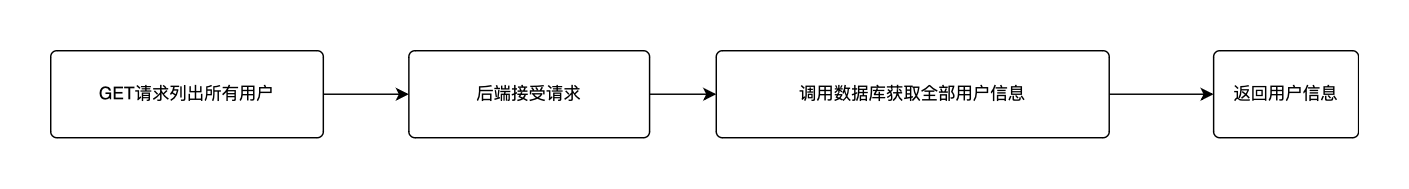
## 4.8获取到达机场天气



## 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。

