****

**阿巴天气预测系统**

**系统实现文档**

**项目Git访问地址：**

[**https://github.com/supertailcat/small\_semester**](https://github.com/Peter-Sundy/software_engineering)

**阿巴阿巴小组**

目录

[1 系统实现 1](#_Toc45571671)

[1.1人员安排 1](#_Toc45571672)

[1.2 登陆注册实现 1](#_Toc45571673)

[1.2.1 数据库实现 1](#_Toc45571674)

[1.2.2 表单和路由实现 1](#_Toc45571675)

[1.3 主站前端模块实现 2](#_Toc45571676)

[1.3.1 前端模块实现简介 2](#_Toc45571677)

[1.3.2 前端模块实现 3](#_Toc45571678)

[1.3.3 前端模块实现界面 4](#_Toc45571679)

[1.4 主站后端模块实现 4](#_Toc45571680)

[1.4.1后端模块实现简介 4](#_Toc45571681)

[1.4.2后端模块实现 4](#_Toc45571682)

[1.5 Flask数据传输模块实现 5](#_Toc45571683)

[1.5.1数据传输模块实现简介 5](#_Toc45571684)

[1.5.2数据传输模块实现 6](#_Toc45571685)

[1.5.3数据传输模块实现界面 6](#_Toc45571686)

[1.6 预测模块实现 7](#_Toc45571687)

[1.6.1预测模块实现简介 7](#_Toc45571688)

[1.6.2请假模块相关类实现 8](#_Toc45571689)

[1.6.3请假模块实现界面 8](#_Toc45571690)

[1.7 实现过程中的需求和设计变动情况说明 8](#_Toc45571691)

[1.8 实现小结 9](#_Toc45571692)

# 1 系统实现

## 1.1人员安排

准时Ding-员工打开平台系统实现小组

组长：孙世卓 -- 搭建系统框架

组员：赵恒 -- 建立数据库表，导入信息，建立员工登录模块

潘智超 -- 建立留言请假模块

吴嘉林 -- 建立管理员模块，完成设计管理员功能

李红甫 – 进行界面设计及美化

## 1.2 登陆注册实现

### 1.2.1 数据库实现

数据库使用SQLite实现，SQLite是一个软件库，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。SQLite是在世界上最广泛部署的SQL 数据库引擎。SQLite源代码不受版权限制，集成在Django代码中（图1.1）。

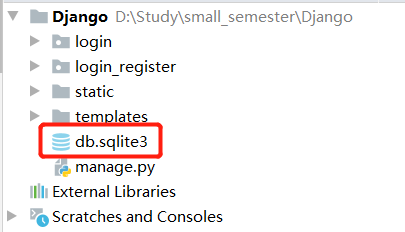


图1.1-数据库实现

### 1.2.2 表单和路由实现

（1）login目录主要包含了普通用户登录系统所需要的操作，属于后端。其中form为表单格式，view为控制逻辑，urls为路由配置。Template保存的为网页内容，static为静态依赖项。网页的部分参数内容会从此加载。

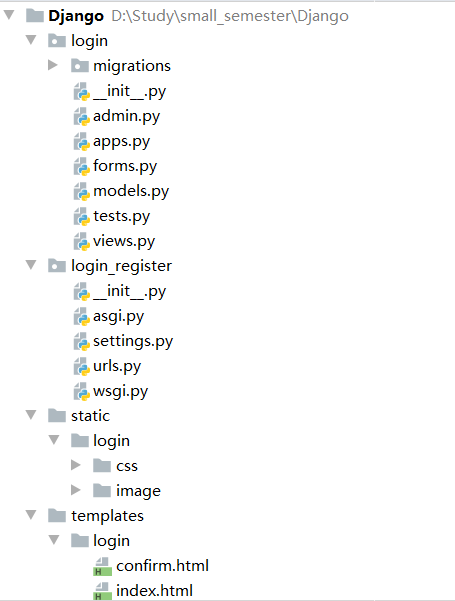


图1.2-MVC

## 1.3 主站前端模块实现

### 1.3.1 前端模块实现简介

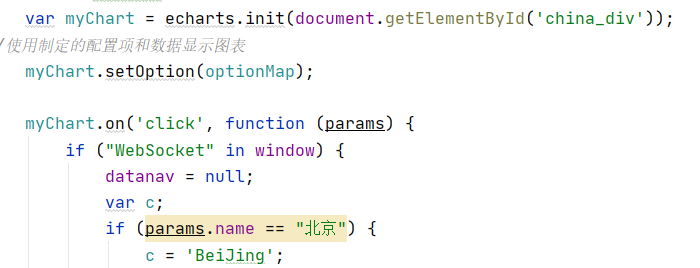
在普通用户进行登录后，主站被呈现给普通用户。前端编写使用了HTML5，和脚本语言JavaScript。HTML5来进行页面布局，标题正文的排版，JavaScript内包含地图控件、图表控件、以及Websocket连接控件。各个控件彼此独立，也有按次序的调用。

### 1.3.2 前端模块实现

Echart实现折线图绘制：



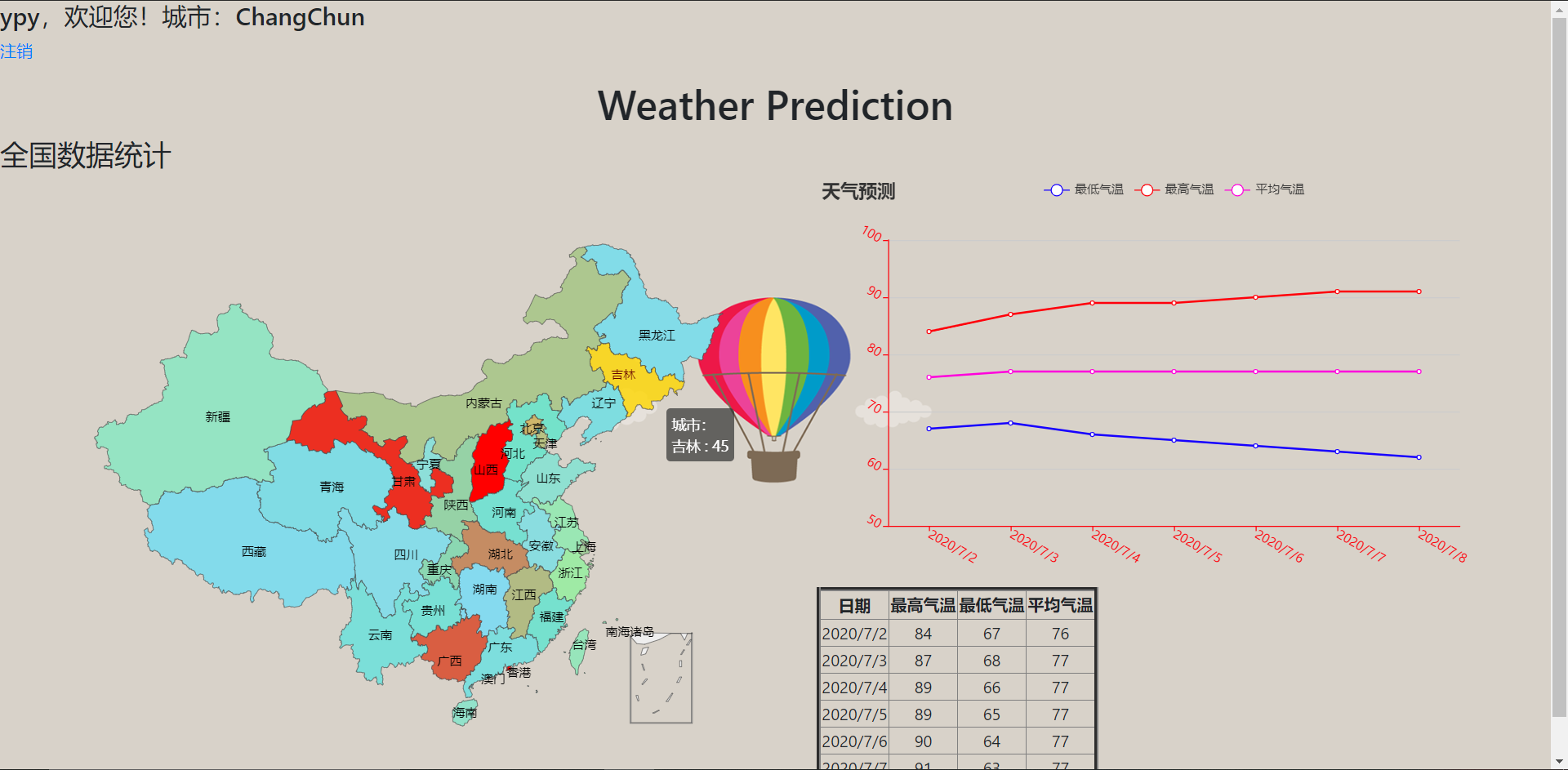
Echart实现地图控件：



Websocket实现数据传输：



### 1.3.3 前端模块实现界面



## 1.4 主站后端模块实现

### 1.4.1后端模块实现简介

主站后端为注销后的跳转，无其它操作。

### 1.4.2后端模块实现

如图所示



## 1.5 Flask数据传输模块实现

### 1.5.1数据传输模块实现简介

通过使用websocket与Django服务器进行数据交互，接受网页服务器发送的请求，读取相应城市的温度预测结果(.json文件)，并发送json数据给网页服务器。

### 1.5.2数据传输模块实现



### 1.5.3数据传输模块实现界面

如图所示。

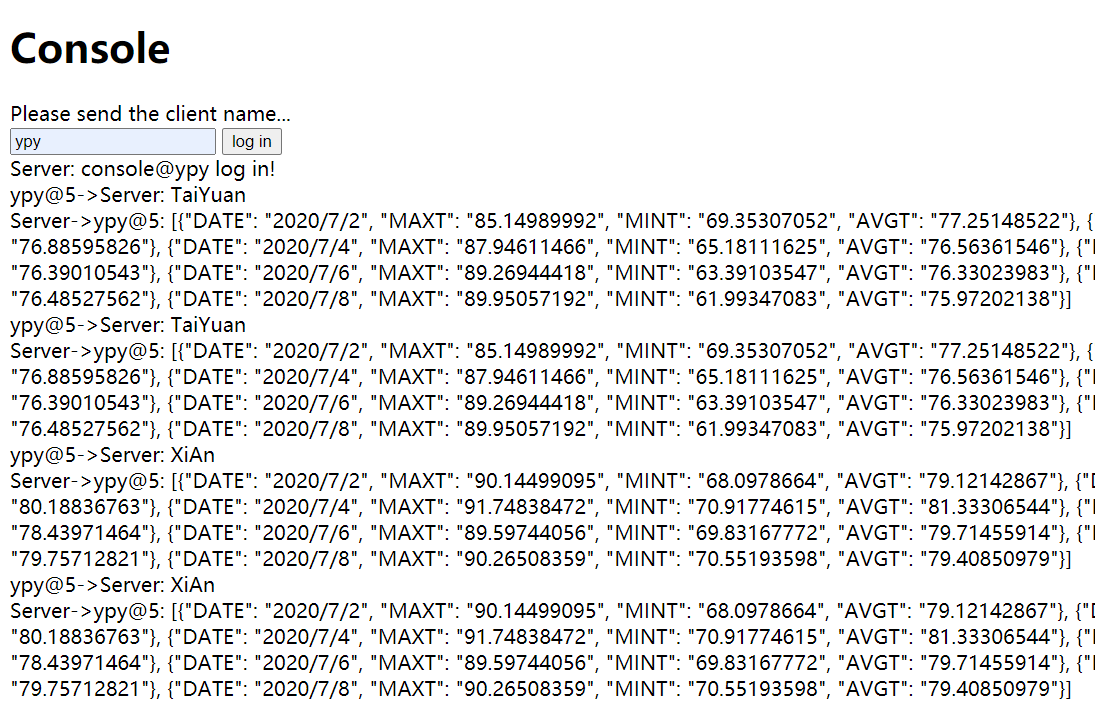


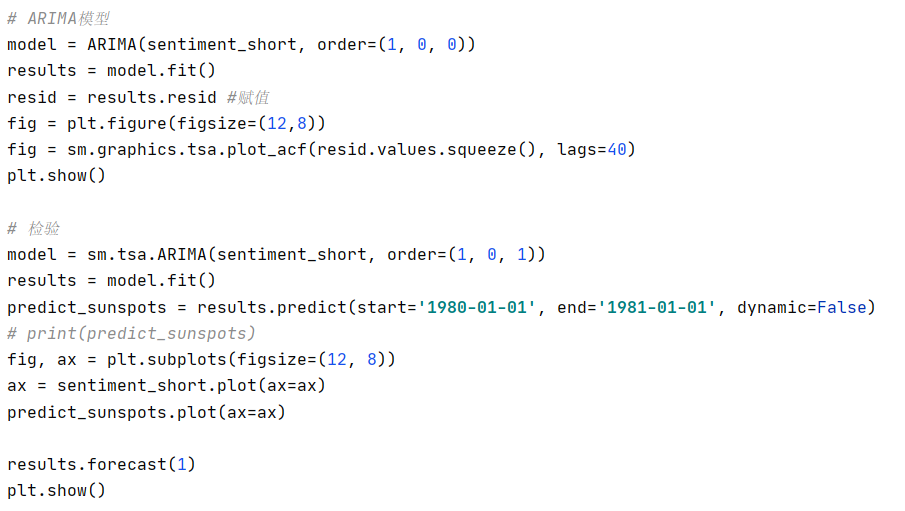
图 Flask控制台

## 1.6 预测模块实现

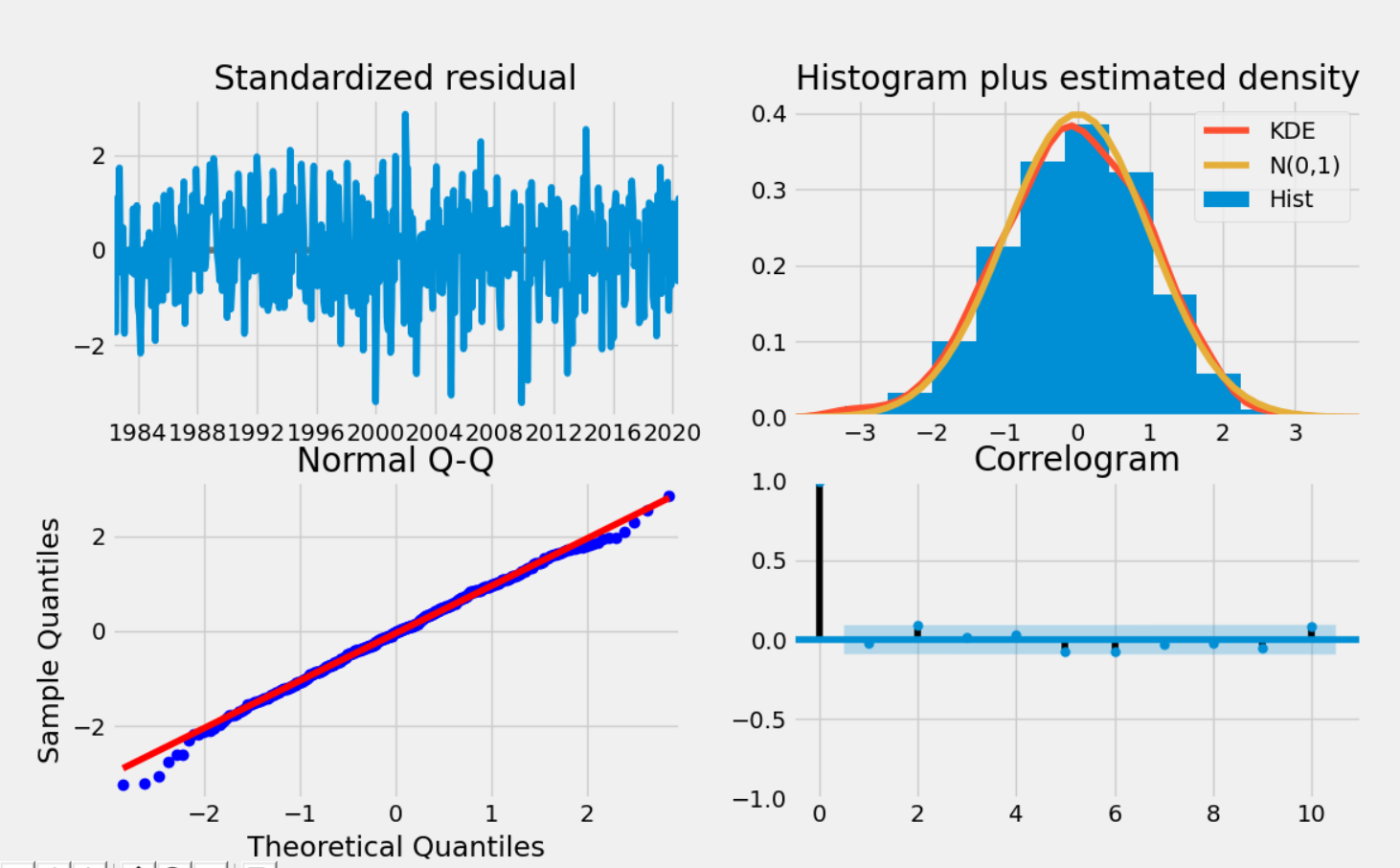
### 1.6.1预测模块实现简介

观测发现气温数据存在明显的周期性变化，因此考虑使SARIMA来进行建模，它可以作为ARIMA的扩展，首先需要去除周期性，去除的方式是在周期间隔上做一次ARIMA，此时可以得到一个非平稳非周期性的时间序列，然后在此基础之上再一次使用ARIMA进行分析。可以表示为：ARIMA(p, d, q) × (P, D, Q)S ，其中P为周期性自回归阶数，D为周期性差分阶数，Q为周期性移动平均阶数，S为周期时间间隔。

### 1.6.2请假模块相关类实现



### 1.6.3请假模块实现界面



## 1.7 实现过程中的需求和设计变动情况说明

在设计过程中对刚开始的虚拟机需求进行了调整。原先是考虑到服务器资源有限，而采用虚拟机模拟Linux环境。但考虑到环境配置繁琐，而且服务器可以方便地取得，便舍弃了虚拟机的设计，采用两台服务器作为Django服务器和Flask服务器。同时，基于这种改动，我们的项目支持任何地点不受限制地使用，更加具有成品性质。

本项目网页服务提供是基于Django的。不过原需求为Tomcat，在经过小组成员的尝试下，发现Tomcat的环境配置并不简单。JDK版本需求与我们本机的JDK版本并不相同，而且降版本有无法安装的风险，考虑到多方面原因，我们决定采用功能相似的Django代替Tomcat，虽然泛用性和能力上，Django不及Tomcat，但完成本次项目，也仅仅发挥Django的一小部分功能就可以了。节省了开发时间，也避免了跳坑。

我们的系统虽然主要服务为天气预测，但是后期在权限管理上也做出了较高的要求。支持部门管理、角色管理、用户管理等操作，基于RBAC和树形结构，权限管理可靠且易于使用。

## 1.8 实现小结

本项目组成员分工明确，积极讨论，在系统代码编写中讨论交流，互相提高了对Python语言的进一步了解。在实验中进行了多轮测试，在不断测试中找出程序的问题所在并加以调试，最终实现一个完整的结合了人工智能、web开发、模块管理的天气预测系统。在编写过程中，我们在最为复杂的Django部分与Flask模块中的路由、websocket的理解以及各模块的接口与调用中产生了不同想法，在进行讨论后一同解决了此问题，最终形成了如泳道图所示的完整系统。主要的功能一应俱全。项目组成员在整个过程中进一步的加深了对软件工程流程的掌握与认知，加强了程序编写能力与纠错调试能力，收获颇丰。在成员的积极努力、共同奋战下，我们系统的实现及测试工作取得了优秀成果。