

SHANGHAI JIAO TONG UNIVERSITY

**课程论文**

## COURSE PAPER



论文题目： 导师论文&评价网设计文档

学生姓名: 夏里宾

学生学号: 518021910503

专 业: 信息安全

指导教师: 黄征

学院(系): 电子信息与电气工程学院

目录

[COURSE PAPER 1](#_Toc70089992)

[1 引言 2](#_Toc70089993)

[1.1. 编写目的 2](#_Toc70089994)

[1.2. 项目概况 3](#_Toc70089995)

[1.3. 参考资料 3](#_Toc70089996)

[2 系统概述 3](#_Toc70089997)

[2.1. 系统体系结构与功能 3](#_Toc70089998)

[2.2. 系统整体架构 3](#_Toc70089999)

[3 具体实现方法 4](#_Toc70090000)

[3.1. 整体框架的搭建 4](#_Toc70090001)

[3.2. 登录系统 5](#_Toc70090002)

[3.3. 论文系统 5](#_Toc70090003)

[3.4. 评论系统 6](#_Toc70090004)

[3.5. 爬虫系统 6](#_Toc70090005)

[4 公用接口程序设计说明 7](#_Toc70090006)

[4.1. 全局变量 7](#_Toc70090007)

[4.2. 公用界面 7](#_Toc70090008)

[4.3. 公用函数和过程 8](#_Toc70090009)

## 1 引言

### 1.1. 编写目的

编写这篇文档主要是为了能够让用户更清楚地了解本网站是如何运作的，让他们能够更放心的去使用。另一个目的是记录我框架配置的全过程，以便作为之后类似项目的参考，而且也能更加有利于网页的进一步优化。

### 1.2. 项目概况

本项目是基于vue和django框架的web界面，名为导师论文&评价网，学生们可以在网页上查找自己感兴趣的导师，并且在网页中可以直接获取导师的论文信息和学生评价情况。

Github：<https://github.com/faderer/professer_paper_comment>

### 1.3. 参考资料

<https://zhuanlan.zhihu.com/p/128976272?utm_source=qq>

<https://docs.djangoproject.com/en/3.1/intro/tutorial01/>

## 2 系统概述

### 2.1. 系统体系结构与功能

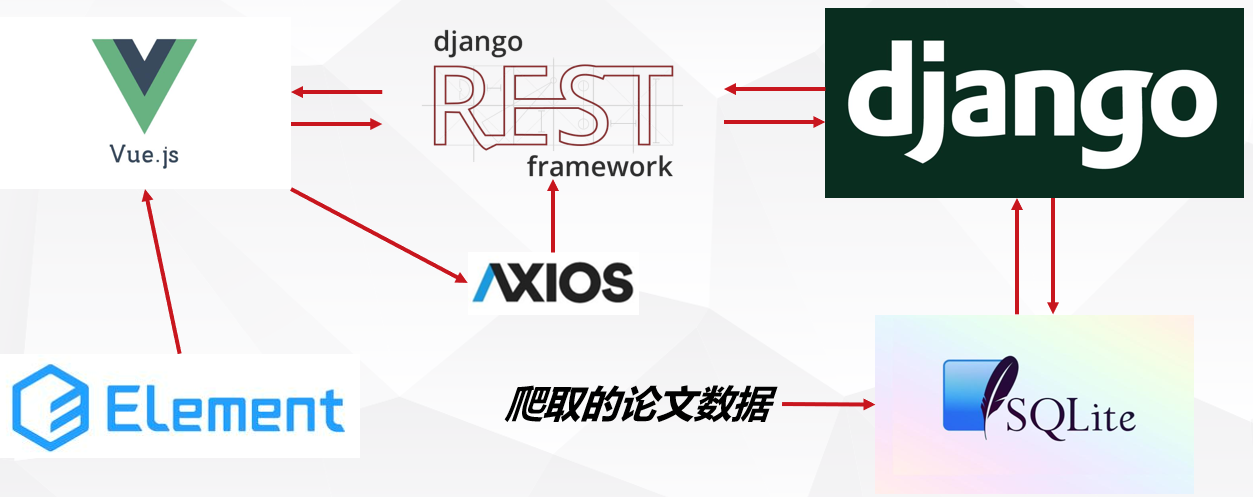
导师评价网分为三个子系统：

登录系统：为了避免外校学生的恶意刷屏，网站中建立了一个登录系统，可以在注册过程中对邮箱的后缀进行合法性检测。

论文系统：论文的爬虫是通过python脚本和chromedriver.exe，传至前端后可供学生在线查阅与下载。

评价系统：已登录的学生可以在导师的界面下进行匿名评价。

### 2.2. 系统整体架构



首先，利用python脚本将爬取的导师信息批量传入到django自带的数据库。然后，利用django rest framework生成一个可导入导出数据的数据库api。跨域配置完成后，vue就可以利用axios模块向DRFapi发送网络请求并获取相应的数据。最后可以利用element组件进行网页的优化。

## 3 具体实现方法

### 3.1. 整体框架的搭建

* 首先安装和配置好vue，django，DRF等必要框架。
* 接着初始化前端与后端：

vue-init webpack webfront //搭建vue脚手架

django-admin startproject mysite

python manage.py startapp polls //由于是第一次尝试django所以project和app的名称都是直接采用教程中的名称，调试完成之后就直接实际使用了，来不及修改，不过也并无大碍。

* 在setting.py文件中的app配置加入rest framework和polls

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

# demo add

'rest\_framework',

'polls',

]

* 在url.py文件中进行整个项目的路由配置，让访问api/路径时候转到books应用中的urls.py文件配置进行处理。

urlpatterns = [

path('admin/', admin.site.urls),

path('api/', include('polls.urls')), # demo add

]

* 前后端跨域配置：

首先在命令行中进行对应模块的安装：

pip install django-cors-headers

在项目中添加该模块：

INSTALLED\_APPS = [

...

# demo

'corsheaders',

...

]

MIDDLEWARE = [

'corsheaders.middleware.CorsMiddleware', # 需注意与其他中间件顺序，这里放在最前面即可

...

]

# 支持跨域配置开始

CORS\_ORIGIN\_ALLOW\_ALL = True

CORS\_ALLOW\_CREDENTIALS = True

* 前端网络请求配置：

首先命令行安装axios模块，如果没有安装cnpm就还是用npm安装：

cnpm install axios

为了方便管理api请求的各种逻辑，在前端项目的src目录下创建api目录，然后创建api.js和index.js文件。index.js文件是对axios做配置（具体代码就不展示了）

* 配置element

推荐使用 npm 的方式安装，它能更好地和 webpack 打包工具配合使用。

npm i element-ui -S

完整引入

在 main.js 中写入以下内容：

import Vue from 'vue';

import ElementUI from 'element-ui';

import 'element-ui/lib/theme-chalk/index.css';

import App from './App.vue';

Vue.use(ElementUI);

new Vue({

el: '#app',

render: h => h(App)

});

至此前后端的框架搭建就算完成了。

### 3.2. 登录系统

为了避免外校学生的恶意刷屏，网站中建立了一个登录系统，可以在注册过程中对邮箱的后缀进行合法性检测。实现方法是注册时将合法数据传入数据库，登录时将账号密码与数据库中的注册信息逐条比较，进而判断用户是否存在。

### 3.3. 论文系统

点击导师头像可进入导师界面，首先是从网安官网获取的导师信息，接下来是导师的顶会论文信息，点击即可在线查看与下载。实现方法是对django中的urls.py进行相对路径的配置，利用serve定位到对应的论文目录。

re\_path(r'paper/(?P<path>.\*)$', serve, {'document\_root': settings.MEDIA\_ROOT})

### 3.4. 评论系统

接下来是评论功能，同样是利用已搭建好的框架进行数据传输，再增设一个全局变量来对输入框的隐藏与显示进行控制。

export default{

    iflog: 0,

    setlog(iflog){

*this*.iflog = iflog;

    }

}

### 3.5. 爬虫系统

论文的爬虫是通过python脚本和chromedriver.exe，外加如下所示的json文件来实现的。在当前json配置下爬取的是导师的A类论文，可以通过修改journal来获取B类或C类论文，也可以调整citation的大小利用引用数来筛选论文。fpath和chromedriver\_path分别对应爬虫目标目录和chromedriver.exe的目录。

{

    "keywords":

    [

        "Chaoping Xing",

        "Dawu Gu",

        "Shengli Liu",

        "Xinghao Jiang",

        "Jianhua Li",

        "Shenghong Li",

        "Yuan Luo",

        "Shuo Shao",

        "Deng Tang",

        "Lei Wang",

        "Chen Yuan",

        "Shuai Han",

        "Shengyun Liu"

    ],

    "journals":

    [

        "IEEE Symposium on Security and Privacy",

        "Usenix  Security Symposium",

        "European Cryptology Conference",

        "ACM Computer and Communications Security Conference",

        "Symposium on Network and Distributed System Security",

        "International Cryptology Conference",

        "IEEE Transactions on Information Forensics and Security",

        "IEEE Transactions on Dependable and Secure Computing",

        "Journal of Cryptology"

    ],

    "citation":0,

    "fpath":

        "C:\\Users\\fader\\Desktop\\test\_crawler\\test2",

    "chromedriver\_path":

        "C:\\Users\\fader\\Desktop\\chromedriver.exe"

}

## 4 公用接口程序设计说明

### 4.1. 全局变量

用来判断是否有用户登录

export default{

    iflog: 0,

    setlog(iflog){

*this*.iflog = iflog;

    }

}

### 4.2. 公用界面

网页的主界面，其余界面都是以组件的形式外加router的控制添加到根界面中的。

<template>

  <div id="app">

    <div class="overall">

    <div class="top">

      <div class="tab-box">

      <span

        v-for="(i,index) in tabs"

        :key="index"

        :class="defaultIndex==index?'active':''"

        @click="defaultIndex=index"

      >

        <div class="dot"></div>

        <router-link :to="i.url" tag="span">

          {{i.text}}

        </router-link>

      </span>

      <div class="bar-box">

        <div class="bar"></div>

      </div>

      </div>

      <div class="tab-box2">

        <span v-if="*this*.GLOBAL.iflog==0" class="defaultIndex==index?'active':''">

        <div class="dot"></div>

        <router-link to="/register" tag="span">

          sign up

        </router-link>

        </span>

        <span v-if="*this*.GLOBAL.iflog==1" class="defaultIndex==index?'active':''" @click="logout">

        <div class="dot"></div>

        <router-link to="/" tag="span">

          sign out

        </router-link>

        </span>

      </div>

    </div>

### 4.3. 公用函数和过程

前端用来操作后端的公用api函数

import axiosInstance from './index'

const axios = axiosInstance

export const getPapers = () => {return axios.get(`http://localhost:8000/api/paper/`)}

export const postPapers = (title, citation, date) => {return axios.post(`http://localhost:8000/api/paper/`, {'title': title, 'citation': citation,'pub\_date':date})}

export const getComments = () => {return axios.get(`http://localhost:8000/api/comment/`)}

export const postComments = (teacher, content, pub\_date) => {return axios.post(`http://localhost:8000/api/comment/`, {'teacher': teacher, 'content': content,'pub\_date':pub\_date})}

export const getUsers = () => {return axios.get(`http://localhost:8000/api/login/`)}

export const postUsers = (username, password, email) => {return axios.post(`http://localhost:8000/api/login/`, {'username': username, 'password': password,'email':email})}