# 快捷键

Alt+shift+f自动整理代码格式

Ctrl+shift+c网页打开查看器

# 后端新建项目:

打开powershell，cd到dwebblog下，输入django-admin startproject dwebsite，创建后端项目。

将dwebsite拖入vs code，在终端中输入 python manage.py startapp blog，创建app。

终端输入python manage.py makemigrations和python manage.py migrate，创建数据库

终端输入python manage.py runserver启动服务，默认发布在8000端口

设计功能和数据库。

制作数据库，前往dwebsite\blog\models.py中制作数据关系

# 前端新建项目：

在终端输入vue ui，在跳出的页面中点击左下角进入项目管理器即http://localhost:8001/project/select，点击创建，进入C:\Users\jjnh\Desktop\dwebblog路径下创建新项目，项目名称：dwebhtml，选择使用git管理，选择vue2，创建项目。启动项目，查看能不能成功运行

# Github管理

学习网站：<https://www.bilibili.com/video/BV1dK411p7RF>

进入vs code点击左侧栏的源代码管理，点击安装git，进入跳出的网址，点击右侧小电视内的下载2.30.0版本git，下载**[64-bit Git for Windows Setup](https://github.com/git-for-windows/git/releases/download/v2.30.0.windows.2/Git-2.30.0.2-64-bit.exe)**，****进入安装，安装在D盘，后面全默认，其中编辑器选择vs code作为默认编辑器。安装后，什么都不要开启，点击完成。在powershell输入git，查看有没有回应以保证安装成功,再在powershell中输入，

  git config --global user.email "997893314@qq.com"

  git config --global user.name "wordbob"

重新打开vscode，点击源代码管理，点击初始化存储库。在vscode登录github账号

进入github官网，并登录。新建一个Repositories，repository name取名dwebblog，描述为学习建立个人博客，点击创建Repositories

推送方法：，在源代码管理部分点击√（提交），点击【...】-【拉取、推送】-【推送到】，点击添加远程储存库，从github上添加，验证信息。输入test，推送到test仓库，master分支。即可在github上看到看到推送的项目

点击vs code左下角master，创建新分支first，在test目录下新建“.gitignore”文件，将不需要上传的文件的相对路径复制到此处，并将反斜杠换成正斜杠，在前面添加正斜杠。再将结果推送到本地的master，再点击推送

# 引入element ui

前往https://element.eleme.cn/#/zh-CN/component/quickstart了解element ui

在vue ui中安装插件：vue-router和vuex

在依赖中安装开发依赖，输入elementUI，安装。

在dwebhtml\src\main.js中输入import ElementUI from 'element-ui';和Vue.use(ElementUI);import 'element-ui/lib/theme-chalk/index.css';

# 编写页面

创建css文件，在main.js文件中引入css，但要保证自己的css是在最下面import './assets/css/mystyle.css'

# 引入组件

在前端的dwebhtml\src\router\index.js下新建一个新的路由，内容包含url路径、名称、组件地址。

在dwebhtml\src\views下新建一个组件

在dwebhtml\src\App.vue下增加一个路由<router-view></router-view>

将界面上的框框放入到Home.vue中

# 引入summernote作为富文本编辑器

Summernote官网https://summernote.org/getting-started/。

首先需要在vue ui内的依赖安装开发依赖jquery、bootstrap、summernote、popper.js、axios。

在AddArticle中加入<div id="summernote">Hello Summernote</div>作为内容方框。导入$的包，import $ from 'jquery'，Script内写入$('#summernote').summernote();

在dwebhtml\src\main.js内引入summernote、popper.js、bootstrap等等。

# 前后端交互

在dwebhtml\src\views\AddArticle.vue内导入qs和axios

import axios from 'axios'

import Qs from 'qs'

先在前端的比如点击事件内写入一个axios请求。

      axios.post('http://127.0.0.1:9000/api/add-article/',Qs.stringify(article\_data))

      .then(res=>{

        console.log(res)

        })

      .catch(err=>{

        console.log(err)

        });

在后端dwebsite\dwebsite\settings.py导入

import os，

ALLOWED\_HOSTS = [‘\*’]，

在已有的INSTALLED\_APPS后面添加

INSTALLED\_APPS =['rest\_framework','corsheaders', 'blog']，

在通用组件CommonMiddleware前加入

MIDDLEWARE = [‘'corsheaders.middleware.CorsMiddleware'’]，

在文件最后添加

MEDIA\_URL = '/upload/'

MEDIA\_ROOT = os.path.join(BASE\_DIR,'upload').replace("\\",'/')

在后端dwebsite\dwebsite\urls.py中导入

from django.conf.urls.static import static

from django.conf import settings

并添加路由和根目录

# 添加路由和根目录

urlpatterns = [

    path('admin/', admin.site.urls),

] + static(settings.MEDIA\_URL,document\_root=settings.MEDIA\_ROOT)

在后端内创建超级用户

PS C:\Users\jjnh\Desktop\dwebblog\dwebsite> python manage.py createsuperuser

Username (leave blank to use 'jjnh'): admin

Email address: 997893314@qq.com

Password:

Password (again):

Superuser created successfully.

# 后端：服务搭建

在后端dwebsite\blog\admin.py导入数据库内容

from blog.models import Article

admin.site.register(Article)

迁移数据库

python manage.py makemigrations blog

python manage.py migrate blog

在后端dwebsite\dwebsite\urls.py引入库

from django.urls import path,include

在blog下新建文件api\_url.py即我们所有url，地址为dwebsite\blog\api\_url.py，写入

from django.urls import path

urlpatterns = [

]

在dwebsite\dwebsite\urls.py内导入新建的url文件

from blog import api\_url

再将新建的api\_url文件内的url作用到本文件中

urlpatterns = [

    path('admin/', admin.site.urls),

    path('api/',include(api\_url))

] + static(settings.MEDIA\_URL,document\_root=settings.MEDIA\_ROOT)

在blog下新建api.py即视图层，地址为dwebsite\blog\api.py

文件内导入response和api\_view，写入视图层函数及其装饰器

from rest\_framework.response import Response

from rest\_framework.decorators import api\_view

# api装饰器

@api\_view(['post'])

def add\_article(request):

    return Response('ok')

在api\_url内引入api文件

from blog import api

urlpatterns = [

    path('add-article/',api.add\_article)

]

在settings文件最后添加跨域请求的配置

CORS\_ALLOW\_CREDENTIALS = True

CORS\_ORIGIN\_ALLOW\_ALL = True

CORS\_ORIGIN\_WHITELIST = (

    'http://127.0.0.1',

    )

CORS\_ALLOW\_METHODS = (

    'DELETE',

    'GET',

    'OPTIONS',

    'PATCH',

    'POST',

    'PUT',

    'VIEW',

    )

CORS\_ALLOW\_HEADERS = (

    'XMLHttpRequest',

    'X\_FILENAME',

    'accept-encoding',

    'authorization',

    'content-type',

    'dnt', 'origin',

    'user-agent',

    'x-csrftoken',

    'x-requested-with',

    'Pragma',

)

# 使用apipost管理Api并书写文档

使用软件apipost，下载地址https://www.apipost.cn/download.html，安装到D盘。

注册后，新建一个项目，进入该项目控制台新建目录，输入前端写的post请求的url，点击发送。

在网页点击F12查看请求的有效载荷复制到Body内，点击美化/校验格式，再点击导出字段，对他们进行描述，重命名此次接口。

点击分享当前接口可以查看当前的接口文档。

# 后端使用爬虫获取前端返回的富文本内容

使用Beautiful Soup，

网站https://beautifulsoup.readthedocs.io/zh\_CN/v4.4.0/

安装Beautiful Soup，使用pip install beautifulsoup4

在api文件内引入beautifulsoup和request

from bs4 import BeautifulSoup

import requests