### urls.py

urlpatterns = [

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

path('addPatient/', user.views.add\_patient),

path('findPatient/', user.views.find\_patient),

path('login/', user.views.login),

path('register/', user.views.register),

path('receive/', user.views.receive\_origin),

path('upload/', user.views.receive\_upload),

path('getListForDoctor/', user.views.get\_photo\_list\_for\_doctor),

path('getListForPatient/', user.views.get\_photo\_list\_for\_patient),

path('deletePicture/', user.views.delete\_picture),

path('revisePictureName/', user.views.revise\_picture\_name),

path('getPatientInfo/', user.views.get\_patient\_info),

path('uploadPatientIcon/', user.views.receive\_patient\_icon),

path('editPatientInfo/', user.views.edit\_patient\_info),

path('getDoctorInfo/', user.views.get\_doctor\_info),

path('uploadDoctorIcon/', user.views.receive\_doctor\_icon),

path('editDoctorInfo/', user.views.edit\_doctor\_info),

path('getPhotoInfo/', user.views.get\_photo\_info),

path('sendComment/', user.views.send\_comment),

path('deleteComment/', user.views.delete\_comment),

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

re\_path(r'^', TemplateView.as\_view(template\_name="index.html")),

]

### views.py

import subprocess

import cv2

import os

import json

import time

from django.core.paginator import Paginator, PageNotAnInteger, EmptyPage

from django.http import JsonResponse, FileResponse, StreamingHttpResponse

from user.models import Doctor, Patient, Photo, Upload, Comment

from django.contrib.auth.hashers import make\_password, check\_password

# BASEURL = "http://localhost:8000/media/test/"

BASEURL = "http://10.251.0.251:8000/media/test/"

def get\_patient\_icon(id):

patient = Patient.objects.get(patient\_id=id)

if patient.patient\_icon is not None and len(patient.patient\_icon) > 0:

return BASEURL + 'patient/' + patient.patient\_icon + '.png'

else:

photo = Photo.objects.filter(photo\_patient\_\_patient\_id=patient.patient\_id)

try:

return BASEURL + photo.last().photo\_savename + '\_promap.png'

except:

return ""

pass

def get\_doctor\_icon(id):

doctor = Doctor.objects.get(doctor\_id=id)

if doctor.doctor\_icon is not None and len(doctor.doctor\_icon) > 0:

return BASEURL + 'doctor/' + doctor.doctor\_icon + '.png'

else:

return ""

def login(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

uname = data\_json.get('username')

pwd = data\_json.get('password')

print(uname)

print(pwd)

user = Doctor.objects.filter(doctor\_username=uname)

if len(user) == 0:

return JsonResponse({'success': False, 'message': '用户不存在'})

if len(user) > 0:

if check\_password(pwd, (user[0]).doctor\_password):

return JsonResponse({'success': True, 'message': '登录成功', 'userID': user[0].doctor\_id,

'role': 'Doctor'})

else:

return JsonResponse({'success': False, 'message': '密码错误'})

else:

return JsonResponse({})

def register(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

new\_name = data\_json.get('username')

temp\_pwd = data\_json.get('password')

new\_email = data\_json.get('email')

new\_ph = data\_json.get('phone')

new\_realname = data\_json.get('realname')

new\_sex = data\_json.get('sex')

print(new\_name)

new\_pwd = make\_password(temp\_pwd, None, 'pbkdf2\_sha1')

print(new\_pwd)

print(new\_email)

print(new\_ph)

print(new\_realname)

print(new\_sex)

if new\_name is None:

return JsonResponse({'success': False, 'message': '未输入'})

else:

space = Doctor.objects.filter(doctor\_username=new\_name)

if len(space) > 0:

return JsonResponse({'success': False, 'message': '用户名已存在'})

else:

new\_d = Doctor()

new\_d.doctor\_username = new\_name

new\_d.doctor\_password = new\_pwd

new\_d.doctor\_email = new\_email

new\_d.doctor\_phone = new\_ph

new\_d.doctor\_realname = new\_realname

if new\_sex == 0 or new\_sex == 1:

new\_d.doctor\_sex = new\_sex

new\_d.save()

return JsonResponse({'success': True, 'message': '注册成功'})

else:

return JsonResponse({})

def get\_photo\_list\_for\_doctor(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

u\_id = data\_json.get('user\_id')

pagenum = data\_json.get('pagenum') # 当前页码数

pagesize = data\_json.get('pagesize') # 每页显示条目数

title = data\_json.get('title') # 每页显示条目数

show\_all = data\_json.get('showAll')

if show\_all:

result = Photo.objects.filter(photo\_realname\_\_contains=title)

else:

if data\_json.get('other\_user\_id') is not None:

result = Photo.objects.filter(photo\_realname\_\_contains=title,

photo\_doctor\_\_doctor\_id=data\_json.get('other\_user\_id'))

else:

result = Photo.objects.filter(photo\_doctor\_\_doctor\_id=u\_id, photo\_realname\_\_contains=title)

apply\_list = list(

result.values('photo\_id', 'photo\_realname', 'photo\_savename'))

total = len(apply\_list)

paginator = Paginator(apply\_list, pagesize)

try:

p\_list = paginator.page(pagenum)

except PageNotAnInteger as e:

p\_list = paginator.page(1) # pagenum不是整数->返回第一页数据

except EmptyPage as e:

if int(pagenum) > paginator.num\_pages:

p\_list = paginator.page(paginator.num\_pages) # pagenum大于页码范围->获取最后一页数据

else:

p\_list = paginator.page(1) # pagenum小于页码范围->获取第一页数据

for photo in p\_list:

patient = Photo.objects.get(photo\_id=photo['photo\_id']).photo\_patient

doctor = Photo.objects.get(photo\_id=photo['photo\_id']).photo\_doctor

photo['patient'] = {

'id': patient.patient\_id,

'name': patient.patient\_name,

'icon': get\_patient\_icon(patient.patient\_id),

'isDoctor': False,

'age': patient.patient\_age,

}

photo['photo\_origin'] = BASEURL + photo['photo\_savename'] + '\_origin.png'

photo['photo\_promap'] = BASEURL + photo['photo\_savename'] + '\_promap.png'

upload = Upload.objects.filter(doctor\_id=u\_id, photo\_id=photo['photo\_id'])

if len(upload) > 0:

photo['photo\_upload'] = BASEURL + upload[0].upload\_savename + '\_upload.png'

else:

photo['photo\_upload'] = ''

if doctor.doctor\_realname is not None and doctor.doctor\_realname != '':

name = doctor.doctor\_realname

else:

name = doctor.doctor\_username

photo['doctor'] = {

'id': doctor.doctor\_id,

'name': name,

'icon': get\_doctor\_icon(doctor.doctor\_id),

'isDoctor': True,

'age': doctor.doctor\_age,

}

return JsonResponse({'success': True, 'message': '返回list成功', 'imageList': list(p\_list), 'total': total},

safe=False, status=200,

json\_dumps\_params={'ensure\_ascii': False})

else:

return JsonResponse({})

def get\_photo\_list\_for\_patient(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

u\_id = data\_json.get('user\_id')

d\_id = data\_json.get('id')

pagenum = data\_json.get('pagenum') # 当前页码数

pagesize = data\_json.get('pagesize') # 每页显示条目数

print(u\_id)

result = Photo.objects.filter(photo\_patient\_\_patient\_id=u\_id)

apply\_list = list(

result.values('photo\_id', 'photo\_realname', 'photo\_savename'))

total = len(apply\_list)

paginator = Paginator(apply\_list, pagesize)

try:

p\_list = paginator.page(pagenum)

except PageNotAnInteger as e:

p\_list = paginator.page(1) # pagenum不是整数->返回第一页数据

except EmptyPage as e:

if int(pagenum) > paginator.num\_pages:

p\_list = paginator.page(paginator.num\_pages) # pagenum大于页码范围->获取最后一页数据

else:

p\_list = paginator.page(1) # pagenum小于页码范围->获取第一页数据

for photo in p\_list:

patient = Photo.objects.get(photo\_id=photo['photo\_id']).photo\_patient

doctor = Photo.objects.get(photo\_id=photo['photo\_id']).photo\_doctor

photo['patient'] = {

'id': patient.patient\_id,

'name': patient.patient\_name,

'icon': get\_patient\_icon(patient.patient\_id),

'isDoctor': False,

'age': patient.patient\_age,

}

photo['photo\_origin'] = BASEURL + photo['photo\_savename'] + '\_origin.png'

photo['photo\_promap'] = BASEURL + photo['photo\_savename'] + '\_promap.png'

upload = Upload.objects.filter(doctor\_id=d\_id, photo\_id=photo['photo\_id'])

if len(upload) > 0:

photo['photo\_upload'] = BASEURL + upload[0].upload\_savename + '\_upload.png'

else:

photo['photo\_upload'] = ''

if doctor.doctor\_realname is not None and doctor.doctor\_realname != '':

name = doctor.doctor\_realname

else:

name = doctor.doctor\_username

photo['doctor'] = {

'id': doctor.doctor\_id,

'name': name,

'icon': get\_doctor\_icon(doctor.doctor\_id),

'isDoctor': True,

'age': doctor.doctor\_age,

}

return JsonResponse({'success': True, 'message': '返回list成功', 'imageList': list(p\_list), 'total': total},

safe=False, status=200,

json\_dumps\_params={'ensure\_ascii': False})

else:

return JsonResponse({})

def get\_photo\_info(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

p\_id = data\_json.get('photo\_id')

d\_id = data\_json.get('id')

photo = Photo.objects.get(photo\_id=p\_id)

return\_object = {}

patient = Photo.objects.get(photo\_id=photo.photo\_id).photo\_patient

doctor = Photo.objects.get(photo\_id=photo.photo\_id).photo\_doctor

myself = Doctor.objects.get(doctor\_id=d\_id)

if myself.doctor\_realname is not None and myself.doctor\_realname != '':

name = myself.doctor\_realname

else:

name = myself.doctor\_username

return\_object['me'] = {

'id': myself.doctor\_id,

'name': name,

'icon': get\_doctor\_icon(myself.doctor\_id),

'isDoctor': True,

'age': myself.doctor\_age,

}

return\_object['patient'] = {

'id': patient.patient\_id,

'name': patient.patient\_name,

'icon': get\_patient\_icon(patient.patient\_id),

'isDoctor': False,

'age': patient.patient\_age,

}

if doctor.doctor\_realname is not None and doctor.doctor\_realname != '':

name = doctor.doctor\_realname

else:

name = doctor.doctor\_username

return\_object['doctor'] = {

'id': doctor.doctor\_id,

'name': name,

'icon': get\_doctor\_icon(doctor.doctor\_id),

'isDoctor': True,

'age': doctor.doctor\_age,

}

return\_object['photo\_id'] = photo.photo\_id

return\_object['photo\_savename'] = photo.photo\_savename

return\_object['photo\_realname'] = photo.photo\_realname

return\_object['photo\_origin'] = BASEURL + photo.photo\_savename + '\_origin.png'

return\_object['photo\_promap'] = BASEURL + photo.photo\_savename + '\_promap.png'

upload\_single = Upload.objects.filter(doctor\_id=d\_id, photo\_id=p\_id)

if len(upload\_single) > 0:

return\_object['photo\_upload'] = BASEURL + upload\_single[0].upload\_savename + '\_upload.png'

else:

return\_object['photo\_upload'] = ''

upload = Upload.objects.filter(photo\_id=photo.photo\_id)

if len(upload) > 0:

upload\_list = []

for single\_upload in upload:

temp\_doctor = single\_upload.doctor

if temp\_doctor.doctor\_realname is not None and temp\_doctor.doctor\_realname != '':

name = temp\_doctor.doctor\_realname

else:

name = temp\_doctor.doctor\_username

upload\_list.append({

'id': single\_upload.upload\_id,

'savename': single\_upload.upload\_savename,

'photo\_upload': BASEURL + single\_upload.upload\_savename + '\_upload.png',

'doctor': {

'id': temp\_doctor.doctor\_id,

'name': name,

'icon': get\_doctor\_icon(temp\_doctor.doctor\_id),

'isDoctor': True,

'age': temp\_doctor.doctor\_age,

}

})

return\_object['photo\_upload\_list'] = upload\_list

else:

return\_object['photo\_upload\_list'] = []

return\_object['comments'] = []

comments = Comment.objects.filter(photo\_id=photo.photo\_id)

for comment in comments:

temp\_doctor = comment.doctor

if temp\_doctor.doctor\_realname is not None and temp\_doctor.doctor\_realname != '':

name = temp\_doctor.doctor\_realname

else:

name = temp\_doctor.doctor\_username

return\_object['comments'].append({

'id': comment.comment\_id,

'content': comment.content,

'doctor': {

'id': temp\_doctor.doctor\_id,

'name': name,

'icon': get\_doctor\_icon(temp\_doctor.doctor\_id),

'isDoctor': True,

'age': temp\_doctor.doctor\_age,

}

})

return JsonResponse({'success': True, 'message': '返回list成功', 'photo': return\_object}, status=200)

else:

return JsonResponse({})

def send\_comment(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

content = data\_json.get('content')

photo\_id = data\_json.get('photo\_id')

doctor\_id = data\_json.get('doctor\_id')

if content is None or photo\_id is None or doctor\_id is None:

return JsonResponse({'success': False, 'message': '参数错误'}, status=400)

comment = Comment(

content=content,

photo\_id=photo\_id,

doctor\_id=doctor\_id

)

comment.save()

return JsonResponse({'success': True, 'message': '评论成功'}, status=200)

else:

return JsonResponse({})

def delete\_comment(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

comment\_id = data\_json.get('comment\_id')

doctor\_id = data\_json.get('doctor\_id')

if comment\_id is None or doctor\_id is None:

return JsonResponse({'success': False, 'message': '参数错误'}, status=400)

comment = Comment.objects.filter(comment\_id=comment\_id, doctor\_id=doctor\_id)

if comment.count() == 0:

return JsonResponse({'success': False, 'message': '评论不存在'}, status=400)

comment.delete()

return JsonResponse({'success': True, 'message': '删除成功'}, status=200)

else:

return JsonResponse({})

def receive\_origin(request):

if request.method == 'POST':

image = request.FILES.get('pic\_img')

u\_id = request.POST.get('user\_id')

p\_id = request.POST.get('id')

obj = Photo()

obj.photo\_doctor = Doctor.objects.get(doctor\_id=u\_id)

print(p\_id)

obj.photo\_patient = Patient.objects.get(patient\_id=p\_id)

base\_dir = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

fn = time.strftime('%Y%m%d%H%M%S')

file\_path = os.path.join(base\_dir, 'media', 'test', str(obj.photo\_id))

with open(file\_path, 'wb+') as f:

for chunk in image.chunks():

f.write(chunk)

img = cv2.imread(file\_path)

new\_path = './media/test/' + str(obj.photo\_patient.patient\_id) + '-' + str(

obj.photo\_doctor.doctor\_id) + '-' + fn + '.png'

cv2.imwrite(new\_path, img, ) # 保存为png

os.remove(file\_path)

obj.photo\_realname = os.path.basename(os.path.splitext(image.name)[0])

obj.photo\_savename = str(obj.photo\_patient.patient\_id) + '-' + str(obj.photo\_doctor.doctor\_id) + '-' + fn

obj.save()

if BASEURL.startswith("http://10.251.0.251"):

subprocess.Popen("python ./test/test.py" + " " + os.path.basename(new\_path), shell=True)

else:

print("dest\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")

subprocess.Popen("python ./test/local\_test.py" + " " + os.path.basename(new\_path), shell=True)

return JsonResponse('message="上传成功', safe=False)

else:

return JsonResponse({})

def receive\_upload(request):

if request.method == 'POST':

image = request.FILES.get('pic\_img')

p\_id = request.POST.get('photo\_id')

d\_id = request.POST.get('id')

photo = Photo.objects.get(photo\_id=p\_id)

doctor = Doctor.objects.get(doctor\_id=d\_id)

base\_dir = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

file\_path = os.path.join(base\_dir, 'media', 'test', photo.photo\_savename)

with open(file\_path, 'wb+') as f:

for chunk in image.chunks():

f.write(chunk)

img = cv2.imread(file\_path)

fn = time.strftime('%Y%m%d%H%M%S')

upload = Upload()

upload.doctor = doctor

upload.photo = photo

upload.upload\_savename = str(photo.photo\_id) + '-' + str(doctor.doctor\_id) + '-' + fn

cv2.imwrite('./media/test/' + upload.upload\_savename + '\_upload.png', img, ) # 保存为png

os.remove(file\_path)

upload.save()

return JsonResponse('message="上传成功', safe=False)

else:

return JsonResponse({})

def delete\_picture(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

photo\_id = data\_json.get('photoID')

print(photo\_id)

p = Photo.objects.filter(photo\_id=photo\_id)

if len(p) == 0:

return JsonResponse({'success': False, 'message': '图片'}, status=404)

else:

photo = Photo.objects.get(photo\_id=photo\_id)

save\_name = photo.photo\_savename

try:

os.remove('./media/test/' + save\_name + '\_promap.png')

os.remove('./media/test/' + save\_name + '\_origin.png')

except FileNotFoundError as e:

print(e)

Comment.objects.filter(photo=photo).delete()

photos = Upload.objects.filter(photo=photo)

for single\_photo in photos:

os.remove('./media/test/' + single\_photo.upload\_savename + '\_upload.png')

photos.delete()

photo.delete()

return JsonResponse({'success': True, 'message': '删除成功'}, status=200)

else:

return JsonResponse({})

def revise\_picture\_name(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

new\_name = data\_json.get('newName')

photo\_id = data\_json.get('photoID')

picture = Photo.objects.filter(photo\_id=photo\_id)

if len(picture) == 0:

return JsonResponse({'success': False, 'message': '不存在该图片'}, status=404)

else:

obj = Photo.objects.get(photo\_id=photo\_id)

obj.photo\_realname = new\_name

obj.save()

return JsonResponse({'success': True, 'message': '修改成功'}, status=200)

else:

return JsonResponse({})

def add\_patient(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

name = data\_json.get('name')

if len(name) == 0:

return JsonResponse({'success': False, 'message': '姓名不能为空'}, status=404)

else:

new\_p = Patient()

new\_p.patient\_name = name

new\_p.save()

return JsonResponse({'success': True, 'message': '添加患者成功', 'patient\_ID': new\_p.patient\_id}, status=200)

else:

return JsonResponse({})

def find\_patient(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

name = data\_json.get('name')

p = Patient.objects.filter(patient\_name\_\_contains=name)

res = []

if len(p) == 0:

# new\_p = Patient()

# new\_p.patient\_name = name

# new\_p.save()

return JsonResponse({'success': False, 'message': '未查询到患者'}, status=200)

else:

for pa in p:

usr = {}

usr['id'] = pa.patient\_id

usr['name'] = pa.patient\_name

usr['isDoctor'] = False

usr['age'] = pa.patient\_age

usr['icon'] = get\_patient\_icon(pa.patient\_id)

res.append(usr)

return JsonResponse({'p\_list': res, 'success': True, 'message': '查询患者成功'}, status=200)

else:

return JsonResponse({})

def get\_patient\_info(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

patient\_id = data\_json.get('id')

p = Patient.objects.filter(patient\_id=patient\_id)

if len(p) == 0:

return JsonResponse({'success': False, 'message': '不存在该患者'}, status=404)

else:

patient = Patient.objects.get(patient\_id=patient\_id)

return JsonResponse(

{'success': True, 'message': '查询成功',

'user': {'name': patient.patient\_name,

'age': patient.patient\_age,

'height': patient.patient\_height,

'weight': patient.patient\_weight,

'sex': patient.patient\_sex,

'email': patient.patient\_email,

'icon': get\_patient\_icon(patient\_id),

'phone': patient.patient\_phone,

'address': patient.patient\_address,

'isDoctor': False,

'id': patient\_id

}

},

status=200)

else:

return JsonResponse({})

def receive\_patient\_icon(request):

if request.method == 'POST':

image = request.FILES.get('pic\_img')

p\_id = request.POST.get('id')

patient = Patient.objects.get(patient\_id=p\_id)

if patient.patient\_icon is not None and patient.patient\_icon != '':

try:

os.remove("./media/test/patient/" + patient.patient\_icon + ".png")

except Exception as e:

pass

base\_dir = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

fn = time.strftime('%Y%m%d%H%M%S')

file\_path = os.path.join(base\_dir, 'media', 'test', 'patient', str(patient.patient\_id))

with open(file\_path, 'wb+') as f:

for chunk in image.chunks():

f.write(chunk)

img = cv2.imread(file\_path)

cv2.imwrite('./media/test/patient/' + str(patient.patient\_id) + '-' + fn + '.png', img, ) # 保存为png

os.remove(file\_path)

patient.patient\_icon = str(patient.patient\_id) + '-' + fn

patient.save()

return JsonResponse('message="上传成功', safe=False)

else:

return JsonResponse({})

def edit\_patient\_info(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

patient\_id = data\_json.get('id')

p = Patient.objects.filter(patient\_id=patient\_id)

if len(p) == 0:

return JsonResponse({'success': False, 'message': '不存在该患者'}, status=404)

else:

patient = Patient.objects.get(patient\_id=patient\_id)

patient.patient\_name = data\_json.get('name')

patient.patient\_age = int(data\_json.get('age'))

patient.patient\_height = int(data\_json.get('height'))

patient.patient\_weight = int(data\_json.get('weight'))

patient.patient\_address = data\_json.get('address')

patient.patient\_phone = data\_json.get('phone')

patient.patient\_email = data\_json.get('email')

if data\_json.get('sex') is not None:

patient.patient\_sex = int(data\_json.get('sex'))

print("??")

patient.save()

return JsonResponse({'success': True, 'message': '修改信息成功'}, status=200)

else:

return JsonResponse({})

def get\_doctor\_info(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

doctor\_id = data\_json.get('id')

p = Doctor.objects.filter(doctor\_id=doctor\_id)

if len(p) == 0:

return JsonResponse({'success': False, 'message': '不存在该医生'}, status=404)

else:

doctor = Doctor.objects.get(doctor\_id=doctor\_id)

return JsonResponse(

{'success': True, 'message': '查询成功',

'user': {

'username': doctor.doctor\_username,

'name': doctor.doctor\_realname,

'age': doctor.doctor\_age,

'sex': doctor.doctor\_sex,

'email': doctor.doctor\_email,

'icon': get\_doctor\_icon(doctor\_id),

'phone': doctor.doctor\_phone,

'isDoctor': True,

'id': doctor\_id

}

},

status=200)

else:

return JsonResponse({})

def receive\_doctor\_icon(request):

if request.method == 'POST':

image = request.FILES.get('pic\_img')

p\_id = request.POST.get('id')

doctor = Doctor.objects.get(doctor\_id=p\_id)

if doctor.doctor\_icon is not None and doctor.doctor\_icon != '':

try:

os.remove("./media/test/doctor/" + doctor.doctor\_icon + ".png")

except Exception as e:

pass

base\_dir = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

fn = time.strftime('%Y%m%d%H%M%S')

file\_path = os.path.join(base\_dir, 'media', 'test', 'doctor', str(doctor.doctor\_id))

with open(file\_path, 'wb+') as f:

for chunk in image.chunks():

f.write(chunk)

img = cv2.imread(file\_path)

cv2.imwrite('./media/test/doctor/' + str(doctor.doctor\_id) + '-' + fn + '.png', img, ) # 保存为png

os.remove(file\_path)

doctor.doctor\_icon = str(doctor.doctor\_id) + '-' + fn

doctor.save()

return JsonResponse('message="上传成功', safe=False)

else:

return JsonResponse({})

def edit\_doctor\_info(request):

if request.method == 'POST':

data\_json = json.loads(request.body)

doctor\_id = data\_json.get('id')

p = Doctor.objects.filter(doctor\_id=doctor\_id)

if len(p) == 0:

return JsonResponse({'success': False, 'message': '不存在该患者'}, status=404)

else:

doctor = Doctor.objects.get(doctor\_id=doctor\_id)

doctor.doctor\_realname = data\_json.get('name')

doctor.doctor\_age = int(data\_json.get('age'))

doctor.doctor\_phone = data\_json.get('phone')

doctor.doctor\_email = data\_json.get('email')

if data\_json.get('sex') is not None:

doctor.doctor\_sex = int(data\_json.get('sex'))

print("??")

doctor.save()

return JsonResponse({'success': True, 'message': '修改信息成功'}, status=200)

else:

return JsonResponse({})

### models.py

from django.db import models

class Comment(models.Model):

comment\_id = models.AutoField(db\_column='comment\_ID', primary\_key=True) # Field name made lowercase.

photo = models.ForeignKey('Photo', models.DO\_NOTHING, db\_column='photo\_ID', blank=True, null=True) # Field name made lowercase.

doctor = models.ForeignKey('Doctor', models.DO\_NOTHING, db\_column='doctor\_ID', blank=True, null=True) # Field name made lowercase.

reply\_to = models.ForeignKey('self', models.DO\_NOTHING, db\_column='reply\_to', blank=True, null=True)

content = models.CharField(max\_length=255, blank=True, null=True)

class Meta:

managed = False

db\_table = 'comment'

class Doctor(models.Model):

doctor\_id = models.AutoField(db\_column='doctor\_ID', primary\_key=True) # Field name made lowercase.

doctor\_username = models.CharField(max\_length=255)

doctor\_password = models.CharField(max\_length=255)

doctor\_realname = models.CharField(max\_length=255, blank=True, null=True)

doctor\_email = models.CharField(max\_length=255, blank=True, null=True)

doctor\_icon = models.CharField(max\_length=255, blank=True, null=True)

doctor\_phone = models.CharField(max\_length=255, blank=True, null=True)

doctor\_sex = models.SmallIntegerField(blank=True, null=True)

doctor\_age = models.IntegerField(blank=True, null=True)

class Meta:

managed = False

db\_table = 'doctor'

class Manager(models.Model):

manager\_id = models.AutoField(db\_column='manager\_ID', primary\_key=True) # Field name made lowercase.

manager\_name = models.CharField(max\_length=20)

manager\_password = models.CharField(max\_length=25)

class Meta:

managed = False

db\_table = 'manager'

class Patient(models.Model):

patient\_id = models.AutoField(db\_column='patient\_ID', primary\_key=True) # Field name made lowercase.

patient\_name = models.CharField(max\_length=30)

patient\_phone = models.CharField(max\_length=11, blank=True, null=True)

patient\_address = models.CharField(max\_length=30, blank=True, null=True)

patient\_email = models.CharField(max\_length=255, blank=True, null=True)

patient\_height = models.IntegerField(blank=True, null=True)

patient\_weight = models.IntegerField(blank=True, null=True)

patient\_age = models.IntegerField(blank=True, null=True)

patient\_sex = models.IntegerField(blank=True, null=True)

patient\_icon = models.CharField(max\_length=255, blank=True, null=True)

class Meta:

managed = False

db\_table = 'patient'

class Photo(models.Model):

photo\_id = models.AutoField(db\_column='photo\_ID', primary\_key=True) # Field name made lowercase.

photo\_realname = models.CharField(max\_length=255)

photo\_doctor = models.ForeignKey(Doctor, models.DO\_NOTHING, db\_column='photo\_doctor\_ID', blank=True, null=True) # Field name made lowercase.

photo\_savename = models.CharField(max\_length=255)

photo\_patient = models.ForeignKey(Patient, models.DO\_NOTHING, db\_column='photo\_patient\_ID', blank=True, null=True) # Field name made lowercase.

class Meta:

managed = False

db\_table = 'photo'

class Upload(models.Model):

upload\_id = models.AutoField(db\_column='upload\_ID', primary\_key=True) # Field name made lowercase.

photo = models.ForeignKey(Photo, models.DO\_NOTHING, db\_column='photo\_ID', blank=True, null=True) # Field name made lowercase.

doctor = models.ForeignKey(Doctor, models.DO\_NOTHING, db\_column='doctor\_ID', blank=True, null=True) # Field name made lowercase.

upload\_savename = models.CharField(max\_length=255, blank=True, null=True)

class Meta:

managed = False

db\_table = 'upload'

### test.py

import os

import sys

import warnings

import cv2

from PIL import Image

from skimage.measure import regionprops

import numpy as np

from skimage.io import imsave

from skimage.util import img\_as\_ubyte

from skimage.transform import resize

import torch

from models.get\_model import get\_arch

from utils.model\_saving\_loading import load\_model

from utils import paired\_transforms\_tv04 as p\_tr

def flip\_ud(tens):

return torch.flip(tens, dims=[1])

def flip\_lr(tens):

return torch.flip(tens, dims=[2])

def flip\_lrud(tens):

return torch.flip(tens, dims=[1, 2])

def create\_pred(model, tens, mask, coords\_crop, original\_sz, tta='no'):

act = torch.sigmoid if model.n\_classes == 1 else torch.nn.Softmax(dim=0)

device = torch.device('cuda' if torch.cuda.is\_available() else 'cpu')

with torch.no\_grad():

logits = model(tens.unsqueeze(dim=0).to(device))

logits = logits.squeeze(dim=0)

pred = act(logits)

if tta != 'no':

with torch.no\_grad():

logits\_lr = model(tens.flip(-1).unsqueeze(dim=0).to(device))

logits\_lr = logits\_lr.squeeze(dim=0).flip(-1)

logits\_ud = model(tens.flip(-2).unsqueeze(dim=0).to(device))

logits\_ud = logits\_ud.squeeze(dim=0).flip(-2)

logits\_lrud = model(tens.flip(-1).flip(-2).unsqueeze(dim=0).to(device))

logits\_lrud = logits\_lrud.squeeze(dim=0).flip(-1).flip(-2)

if tta == 'from\_logits':

mean\_logits = torch.mean(torch.stack([logits, logits\_lr, logits\_ud, logits\_lrud]), dim=0)

pred = act(mean\_logits)

elif tta == 'from\_preds':

pred\_lr = act(logits\_lr)

pred\_ud = act(logits\_ud)

pred\_lrud = act(logits\_lrud)

pred = torch.mean(torch.stack([pred, pred\_lr, pred\_ud, pred\_lrud]), dim=0)

else:

raise NotImplementedError

pred = pred.detach().cpu().numpy()[-1] # this takes last channel in multi-class, ok for 2-class

# Orders: 0: NN, 1: Bilinear(default), 2: Biquadratic, 3: Bicubic, 4: Biquartic, 5: Biquintic

pred = resize(pred, output\_shape=original\_sz, order=3)

full\_pred = np.zeros\_like(mask, dtype=float)

full\_pred[coords\_crop[0]:coords\_crop[2], coords\_crop[1]:coords\_crop[3]] = pred

#full\_pred[~mask.astype(bool)] = 0

return full\_pred

def save\_pred(full\_pred, im\_name):

# im\_name = im\_name.rsplit('/', 1)[-1]

save\_name = 'media/test/' + os.path.splitext(sys.argv[1])[0] + '\_promap.png'

with warnings.catch\_warnings():

warnings.simplefilter("ignore")

# this casts preds to int, loses precision but meh

imsave(save\_name, img\_as\_ubyte(full\_pred))

def crop\_to\_fov(img, mask):

img\_g = np.array(img)

h,w,c = img\_g.shape

for i in range(0,h):

summ = np.sum(img\_g[i,:,:])

if summ!=0:

up\_r = i

break

for i in range(0,w):

summ = np.sum(img\_g[:,i,:])

if summ!=0:

left\_r = i

break

for i in range(h-1,-1,-1):

summ = np.sum(img\_g[i,:,:])

if summ!=0:

down\_r = i

break

for i in range(w-1,-1,-1):

summ = np.sum(img\_g[:,i,:])

if summ!=0:

right\_r = i

break

new\_img = img\_g[up\_r:down\_r,left\_r:right\_r]

im\_crop = Image.fromarray(new\_img)

return im\_crop, [up\_r, left\_r, down\_r, right\_r]

def test():

os.environ['CUDA\_VISIBLE\_DEVICES'] = '1,0,3,2'

device = torch.device("cuda")

im\_name = 'media/test/'+sys.argv[1]

img = cv2.imread(im\_name)

im\_name = 'media/test/' + os.path.splitext(sys.argv[1])[0] + '\_origin.png'

cv2.imwrite(im\_name, img)

print(im\_name + ' saved')

experiment\_path = '/home/zy/code/experiments/drive'

config\_file = "/home/zy/code/experiments/drive/config.cfg"

im\_size = (512, 512)

tg\_size = (512, 512)

model\_name = "unet"

print('\* Instantiating model = ' + model\_name)

model = get\_arch(model\_name).to(device)

print('\* Loading trained weights...')

try:

model, stats = load\_model(model, experiment\_path, device)

except RuntimeError:

sys.exit('---- bad config specification (check layers, n\_classes, etc.) ---- ')

model.eval()

print('\* Start predicting...')

image = Image.open(im\_name)

mask = np.zeros\_like(np.array(image)[:,:,1], dtype=float)

image, coords\_crop = crop\_to\_fov(image, mask)

# in numpy convention

original\_sz = image.size[1], image.size[0]

rsz = p\_tr.Resize((512, 512))

tnsr = p\_tr.ToTensor()

tr = p\_tr.Compose([rsz, tnsr])

# only transform image

image = tr(image)

mask = np.array(mask).astype(bool)

full\_pred = create\_pred(model, image, mask, coords\_crop, original\_sz)

print('\* Saving predictions...')

save\_pred(full\_pred, im\_name)

print('\* Done')

if \_\_name\_\_ == '\_\_main\_\_':

test()

### router/index.js

import { createRouter, createWebHistory } from "vue-router";

const routes = [

/\* 公共路由\*/

{

/\*用于指定初始界面的路由 \*/

path: "/",

redirect: "/login",

},

{

path: "/register",

name: "Register",

component: () =>

import ("../views/Public/Register"),

},

{

path: "/login",

name: "Login",

component: () =>

import (

/\* webpackChunkName: "login" \*/

"../views/Public/Login"

),

},

{

path: "/home",

name: "Home",

component: () =>

import (

/\* webpackChunkName: "login" \*/

"../views/Public/Home"

),

},

{

path: "/patient",

name: "Patient",

component: () =>

import (

/\* webpackChunkName: "login" \*/

"../views/User/Patient"

),

},

{

path: "/photo",

name: "Photo",

component: () =>

import (

/\* webpackChunkName: "login" \*/

"../views/Photo"

),

},

{

path: "/doctor/info",

name: "Doctor/Info",

component: () =>

import (

/\* webpackChunkName: "login" \*/

"../views/User/Doctor"

),

},

{

path: "/doctor/user",

name: "Doctor/User",

component: () =>

import ("../views/Doctor/User"),

},

/\* 医生的页面的路由\*/

{

path: "/doctor",

name: "Doctor",

children: [{

path: "upload",

name: "/doctor/upload",

meta: {

title: "上传",

},

component: () =>

import ("../views/Doctor/Upload"),

}, ],

component: () =>

import (

/\* webpackChunkName: "login" \*/

"../views/Doctor/Home"

),

},

/\* 管理员的页面的路由\*/

{

path: "/administrator",

name: "Administrator",

children: [{

path: "user",

name: "user",

meta: {

title: "个人中心",

},

component: () =>

import ("../views/Administrator/User"),

},

{

path: "admin/patient",

name: "admin/patient",

meta: {

title: "患者管理",

},

component: () =>

import ("../views/Administrator/Patient"),

},

{

path: "admin/doctor",

name: "admin/doctor",

meta: {

title: "医生管理",

},

component: () =>

import ("../views/Administrator/Doctor"),

},

{

path: "admin/drug",

name: "admin/drug",

meta: {

title: "药品管理",

},

component: () =>

import ("../views/Administrator/Drug"),

},

{

path: "admin/operation",

name: "admin/operation",

meta: {

title: "手术管理",

},

component: () =>

import ("../views/Administrator/Operation"),

},

],

component: () =>

import (

/\* webpackChunkName: "login" \*/

"../views/Administrator/Home"

),

},

];

const router = createRouter({

history: createWebHistory("/"),

routes,

});

export default router;

### store/index.js

import { createStore } from 'vuex'

import createPersistedState from "vuex-persistedstate"

export default createStore({

state: {

tagsList: [],

collapse: false,

islogin: false,

isAdministrator: false,

username: "未登录",

user\_id: "",

role: 2,

},

mutations: {

login(state, username) {

state.islogin = true

state.username = username

},

changeName(state,username){

state.username = username

},

storeId(state, userid) {

state.user\_id = userid

},

loginout(state) {

state.islogin = false

state.username = "未登录"

state.user\_id = ""

state.tagsList=[]

},

setRole(state, role) {

state.role = role

},

delTagsItem(state, data) {

state

.tagsList

.splice(data.index, 1);

},

setTagsItem(state, data) {

state

.tagsList

.push(data)

},

clearTags(state) {

state.tagsList = []

},

closeTagsOther(state, data) {

state.tagsList = data;

},

closeCurrentTag(state, data) {

for (let i = 0, len = state.tagsList.length; i < len; i++) {

const item = state.tagsList[i];

if (item.path === data.$route.fullPath) {

if (i < len - 1) {

data

.$router

.push(state.tagsList[i + 1].path);

} else if (i > 0) {

data

.$router

.push(state.tagsList[i - 1].path);

} else {

data

.$router

.push("/");

}

state

.tagsList

.splice(i, 1);

break;

}

}

},

// 侧边栏折叠

hadndleCollapse(state, data) {

state.collapse = data;

}

},

actions: {},

modules: {},

plugins: [createPersistedState()]

})

### axios.js

"use strict";

import axios from "axios";

// Full config: https://github.com/axios/axios#request-config

// axios.defaults.baseURL = process.env.baseURL || process.env.apiUrl || ''

// axios.defaults.headers.common['Authorization'] = AUTH\_TOKEN

// axios.defaults.headers.post['Content-Type'] = 'application/x-www-form-urlencoded'

let config = {

//baseURL: "http://10.251.0.251:8000/",

baseURL: "http://10.251.0.251:8000/",

timeout: 60 \* 1000, // Timeout

withCredentials: true, // Check cross-site Access-Control

};

// 创建Axios对象

const Axios = axios.create(config);

// 请求拦截器

Axios.interceptors.request.use(

function(config) {

// Do something before request is sent

return config;

},

function(error) {

// Do something with request error

return Promise.reject(error);

}

);

// 响应拦截器

Axios.interceptors.response.use(

function(response) {

// Do something with response data

return response;

},

function(error) {

// Do something with response error

return Promise.reject(error);

}

);

export default (app) => {

// 挂载axios到Vue对象

app.config.globalProperties.$http = Axios;

};

### request.js

import axios from 'axios';

const service = axios.create({

// process.env.NODE\_ENV === 'development' 来判断是否开发环境

// easy-mock服务挂了，暂时不使用了

// baseURL: 'https://www.easy-mock.com/mock/592501a391470c0ac1fab128',

timeout: 5000

});

service.interceptors.request.use(

config => {

return config;

},

error => {

console.log(error);

return Promise.reject();

}

);

service.interceptors.response.use(

response => {

if (response.status === 200) {

return response.data;

} else {

Promise.reject();

}

},

error => {

console.log(error);

return Promise.reject();

}

);

export default service;

### main.js

import { createApp } from 'vue'

import App from './App.vue'

import router from './router'

import store from './store'

import installElementPlus from './plugins/element'

import './assets/css/icon.css'

import installAxios from './plugins/axios'

import \* as echarts from 'echarts'

const app = createApp(App)

installElementPlus(app)

installAxios(app)

app.config.globalProperties.$echarts = echarts

app

.use(store)

.use(router)

.mount('#app')

### index.html

<!DOCTYPE html>

<html lang="">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width,initial-scale=1.0">

<title>眼底血管分割云平台</title>

<link rel="stylesheet" href="https://at.alicdn.com/t/font\_830376\_qzecyukz0s.css">

</head>

<body>

<noscript>

<strong>We're sorry but <%= htmlWebpackPlugin.options.title %> doesn't work properly without JavaScript enabled.

Please enable it to continue.</strong>

</noscript>

<div id="app"></div>

<!-- built files will be auto injected -->

</body>

</html>

### App.vue

<template>

<keep-alive>

<router-view v-if="$route.meta.keepAlive" />

</keep-alive>

<router-view v-if="!$route.meta.keepAlive" />

</template>

<script>

export default {};

</script>

<style>

@import "./assets/css/main.css";

@import "./assets/css/color-dark.css";

</style>

### color-dark.css

.header {

background-color: #242f42;

}

.login-wrap {

background: #324157;

}

.plugins-tips {

background: #eef1f6;

}

.plugins-tips a {

color: #20a0ff;

}

.el-upload--text em {

color: #20a0ff;

}

.pure-button {

background: #20a0ff;

}

.tags-li.active {

border: 1px solid #409eff;

background-color: #409eff;

}

.message-title {

color: #20a0ff;

}

.collapse-btn:hover {

background: rgb(40, 52, 70);

}

### main.css

\* {

margin: 0;

padding: 0;

}

html,

body,

#app,

.wrapper {

width: 100%;

height: 100%;

overflow: hidden;

}

body {

font-family: 'PingFang SC', "Helvetica Neue", Helvetica, "microsoft yahei", arial, STHeiTi, sans-serif;

}

a {

text-decoration: none

}

.content {

width: auto;

height: calc(100vh - 70px);

padding: 10px;

background: #f0f0f0;

overflow-y: scroll;

box-sizing: border-box;

}

.content-collapse {

left: 65px;

}

.container {

padding: 30px;

background: #fff;

border: 1px solid #ddd;

border-radius: 5px;

}

.crumbs {

margin: 10px 0;

}

.el-table th {

background-color: #f5f7fa !important;

}

.pagination {

margin: 20px 0;

text-align: right;

}

.plugins-tips {

padding: 20px 10px;

margin-bottom: 20px;

}

.el-button+.el-tooltip {

margin-left: 10px;

}

.el-table tr:hover {

background: #f6faff;

}

.mgb20 {

margin-bottom: 20px;

}

.move-enter-active,

.move-leave-active {

transition: opacity .1s ease;

}

.move-enter-from,

.move-leave-to {

opacity: 0;

}

/\*BaseForm\*/

.form-box {

width: 600px;

}

.form-box .line {

text-align: center;

}

.el-time-panel\_\_content::after,

.el-time-panel\_\_content::before {

margin-top: -7px;

}

.el-time-spinner\_\_wrapper .el-scrollbar\_\_wrap:not(.el-scrollbar\_\_wrap--hidden-default) {

padding-bottom: 0;

}

/\*Upload\*/

.pure-button {

width: 150px;

height: 40px;

line-height: 40px;

text-align: center;

color: #fff;

border-radius: 3px;

}

.g-core-image-corp-container .info-aside {

height: 45px;

}

.el-upload--text {

background-color: #fff;

border: 1px dashed #d9d9d9;

border-radius: 6px;

box-sizing: border-box;

width: 360px;

height: 180px;

text-align: center;

cursor: pointer;

position: relative;

overflow: hidden;

}

.el-upload--text .el-icon-upload {

font-size: 67px;

color: #97a8be;

margin: 40px 0 16px;

line-height: 50px;

}

.el-upload--text {

color: #97a8be;

font-size: 14px;

text-align: center;

}

.el-upload--text em {

font-style: normal;

}

/\*VueEditor\*/

.ql-container {

min-height: 400px;

}

.ql-snow .ql-tooltip {

transform: translateX(117.5px) translateY(10px) !important;

}

.editor-btn {

margin-top: 20px;

}

/\*markdown\*/

.v-note-wrapper .v-note-panel {

min-height: 500px;

}

### header.vue

<template>

<div class="header">

<!-- 折叠按钮 -->

<div class="logo">眼底血管分割云平台</div>

<div class="header-right">

<div class="header-user-con">

<!-- 消息中心 -->

<!-- <div class="btn-bell">

<el-tooltip effect="dark" placement="bottom">

<router-link to="/Notification">

<i class="el-icon-bell"></i>

</router-link>

</el-tooltip>

</div> -->

<!-- 用户头像 -->

<div class="user-avator">

<router-link to="/doctor/user"> </router-link>

</div>

<!-- 用户名下拉菜单 -->

<el-dropdown class="user-name" trigger="click" @command="handleCommand">

<span class="el-dropdown-link">

{{ username }}

<i class="el-icon-caret-bottom"></i>

</span>

<template #dropdown>

<el-dropdown-menu v-if="islogin">

<router-link to="/doctor/user" target="\_blank">

<el-dropdown-item>个人信息</el-dropdown-item>

</router-link>

<router-link to="/doctor/upload">

<el-dropdown-item >图像处理</el-dropdown-item>

</router-link>

<el-dropdown-item command="loginout"

>退出登录</el-dropdown-item

>

</el-dropdown-menu>

<el-dropdown-menu v-else>

<router-link to="/login">

<el-dropdown-item>前往登录</el-dropdown-item>

</router-link>

</el-dropdown-menu>

</template>

</el-dropdown>

</div>

</div>

</div>

</template>

<script>

export default {

data() {

return {

fullscreen: false,

name: "linxin",

message: 2,

};

},

computed: {

username() {

let username = this.$store.state.username;

return username;

// return username ? username : this.name;

},

collapse() {

return this.$store.state.collapse;

},

islogin() {

return this.$store.state.islogin;

},

},

methods: {

// 用户名下拉菜单选择事件

handleCommand(command) {

if (command == "loginout") {

this.$store.commit("loginout");

this.$router.push("/login");

}

},

// 侧边栏折叠

collapseChage() {

this.$store.commit("hadndleCollapse", !this.collapse);

},

},

mounted() {

if (document.body.clientWidth < 1500) {

this.collapseChage();

}

},

};

</script>

<style scoped>

.header {

position: relative;

box-sizing: border-box;

width: 100%;

height: 70px;

font-size: 22px;

color: #fff;

}

.collapse-btn {

float: left;

padding: 0 21px;

cursor: pointer;

line-height: 70px;

}

.header .logo {

float: left;

width: 250px;

margin-left: 50px;

line-height: 70px;

}

.header-right {

float: right;

padding-right: 50px;

}

.header-user-con {

display: flex;

height: 70px;

align-items: center;

}

.btn-fullscreen {

transform: rotate(45deg);

margin-right: 5px;

font-size: 24px;

}

.btn-bell,

.btn-fullscreen {

position: relative;

width: 30px;

height: 30px;

text-align: center;

border-radius: 15px;

cursor: pointer;

}

.btn-bell-badge {

position: absolute;

right: 0;

top: -2px;

width: 8px;

height: 8px;

border-radius: 4px;

background: #f56c6c;

color: #fff;

}

.btn-bell .el-icon-bell {

color: #fff;

}

.user-name {

margin-left: 10px;

}

.user-avator {

margin-left: 20px;

}

.user-avator img {

display: block;

width: 40px;

height: 40px;

border-radius: 50%;

}

.el-dropdown-link {

color: #fff;

cursor: pointer;

}

.el-dropdown-menu\_\_item {

text-align: center;

}

</style>

### Home/index.vue

<template>

<div class="about">

<v-header />

<div class="content">

<router-view v-slot="{ Component }">

<transition name="move" mode="out-in">

<keep-alive :include="tagsList">

<component :is="Component" />

</keep-alive>

</transition>

</router-view>

<!-- <el-backtop target=".content"></el-backtop> -->

</div>

</div>

</template>

<script>

import vHeader from "../../../components/Doctor/Header";

import vSidebar from "../../../components/Doctor/Sidebar";

export default {

components: {

vHeader,

vSidebar,

},

computed: {

tagsList() {

return this.$store.state.tagsList.map((item) => item.name);

},

collapse() {

return this.$store.state.collapse;

},

},

};

</script>

### Upload/index.vue

<template>

<div>

<div class="body">

<!-- 左侧的固定快捷按钮 -->

<div class="left">

<upload-left

:props="props"

@downloadAllImage="downloadAllImage"

@deleteAllImage="deleteAllImage"

@getImageList="getImageList"

/>

</div>

<!-- 中间主体部分 -->

<div class="middle">

<div class="middle-container">

<upload-middle

:props="props"

@downloadAllImage="downloadAllImage"

@deleteAllImage="deleteAllImage"

@getImageList="getImageList"

/>

</div>

</div>

<!-- 右边的分页 -->

<div class="right">

<div style="margin:0 auto;text-align:center">

<div style="margin:0 auto;text-align:center">

<el-select

v-model="props.pagesize"

placeholder="Select"

style="width:100px;margin-left:5px"

@change="handleSizeChange(props.pagesize)"

>

<el-option label="1条/页" :value="1"> </el-option>

<el-option label="3条/页" :value="3"> </el-option>

<el-option label="5条/页" :value="5"> </el-option>

<el-option label="10条/页" :value="10"> </el-option>

</el-select>

</div>

<div

style="padding-left:5px;inline-block;margin:0 auto;text-align:center"

>

<el-button-group>

<el-button

style="width:50px; inline-block;"

icon="el-icon-arrow-left"

:disabled="props.pagenum <= 1"

@click="handleCurrentChange(props.pagenum - 1)"

></el-button>

<el-button

style="width:50px; inline-block;"

icon="el-icon-arrow-right"

:disabled="props.pagenum >= props.total / props.pagesize"

@click="handleCurrentChange(props.pagenum + 1)"

>

</el-button>

</el-button-group>

</div>

<div style="width:100px;margin:0 auto;text-align:center">

<el-pagination

:currentPage="props.pagenum"

:page-size="props.pagesize"

layout="pager"

:total="props.total"

@current-change="handleCurrentChange"

>

</el-pagination>

</div>

</div>

</div>

</div>

<el-dialog v-model="props.visible" title="图片上传" width="1200px">

<my-uploader @getImageList="getImageList" :props="props" />

</el-dialog>

</div>

</template>

<script>

import { ElMessageBox, ElMessage } from "element-plus";

import UploadMiddle from "./components/Middle.vue";

import UploadLeft from "./components/Left.vue";

import MyUploader from "./components/Uploader.vue";

export default {

name: "Upload",

components: {

UploadLeft,

UploadMiddle,

MyUploader,

},

data() {

return {

props: {

loadingNewPicture: false,

pagenum: 1,

pagesize: 5,

searchTitle: "",

total: 0,

imageList: [],

visible: false,

showAll:false,

},

};

},

created() {

this.getImageList();

},

methods: {

handleSizeChange(size) {

this.props.pagesize = size;

this.getImageList();

},

handleCurrentChange(current) {

this.props.pagenum = current;

this.getImageList();

},

downloadAllImage() {

var i, len;

for (i = 0, len = this.props.imageList.length; i < len; i++) {

this.downloadASetOfImage(this.props.imageList[i]);

}

},

downloadASetOfImage(singleImage) {

this.downloadIamge(

singleImage.photo\_origin,

singleImage.photo\_realname + "\_origin"

);

this.downloadIamge(

singleImage.photo\_upload,

singleImage.photo\_realname + "\_upload"

);

this.downloadIamge(

singleImage.photo\_promap,

singleImage.photo\_realname + "\_promap"

);

},

deleteAllImage() {

ElMessageBox.confirm(

"本操作无法撤回，您确认要清空本页图片吗？",

"Warning",

{

confirmButtonText: "OK",

cancelButtonText: "Cancel",

type: "warning",

}

)

.then(() => {

var i, len;

for (i = 0, len = this.props.imageList.length; i < len; i++) {

if (

this.props.imageList[i].doctor.id == this.$store.state.user\_id

) {

this.deleteASetOfImage(this.props.imageList[i]);

}

}

ElMessage({

type: "success",

message: "删除成功",

});

})

.catch(() => {

ElMessage({

type: "info",

message: "删除被取消",

});

});

},

async deleteASetOfImage(singleImage) {

if (singleImage.doctor.id != this.$store.state.user\_id) {

return;

}

await new Promise((resolve) => {

this.$http

.post(

"/deletePicture/",

JSON.stringify({ photoID: singleImage.photo\_id })

)

.then((res) => {

{

if (res.data.message === "删除成功") {

this.getImageList();

} else {

this.$message.error("删除失败");

}

}

});

resolve();

}).then(() => {});

},

async getImageList() {

this.props.loadingNewPicture = true;

await new Promise((resolve) => {

this.$http

.post(

"/getListForDoctor/",

JSON.stringify({

title: this.props.searchTitle,

user\_id: this.$store.state.user\_id,

pagenum: this.props.pagenum,

pagesize: this.props.pagesize,

showAll:this.props.showAll,

})

)

.then((res) => {

{

if (res.data.message === "返回list成功") {

this.props.imageList = res.data.imageList;

this.props.total = res.data.total;

} else {

this.$message.error("获取列表失败");

}

this.props.loadingNewPicture = false;

}

});

resolve();

}).then(() => {});

},

downloadIamge(imgsrc, name) {

//下载图片地址和图片名

let image = new Image();

// 解决跨域 Canvas 污染问题

image.setAttribute("crossOrigin", "anonymous");

image.onload = function() {

let canvas = document.createElement("canvas");

canvas.width = image.width;

canvas.height = image.height;

let context = canvas.getContext("2d");

context.drawImage(image, 0, 0, image.width, image.height);

let url = canvas.toDataURL("image/png"); //得到图片的base64编码数据

let a = document.createElement("a"); // 生成一个a元素

let event = new MouseEvent("click"); // 创建一个单击事件

a.download = name || "photo"; // 设置图片名称

a.href = url; // 将生成的URL设置为a.href属性

a.dispatchEvent(event); // 触发a的单击事件

};

image.src = imgsrc;

},

},

};

</script>

<style scope>

@import "style.css";

</style>

### Upload/style.css

#img {

font-size: 15px;

border-radius: 25px;

border: 2px solid #004b97;

padding: 20px;

width: 500px;

height: 50px;

}

.inline-block {

display: inline-block;

margin-right: 30px;

}

.card-header {

display: flex;

justify-content: space-between;

align-items: center;

}

.el-pager {

display: flex;

flex-direction: column;

}

.el-pager>li {

width: 96px;

}

.left {

width: calc(50vw - 500px);

position: absolute;

top: 70;

left: 0;

}

.middle {

width: 1000px;

margin-left: calc(50vw - 500px);

float: left;

}

.middle-container {

width: 100%;

margin: 0 auto;

}

.right {

position: absolute;

top: 70;

right: 0;

width: calc(50vw - 500px);

}

.body {

width: 100%;

min-width: 1200px;

}

### info.vue

<template>

<div>

<div class="prop-name">真实姓名</div>

<div class="prop-value">

{{ thisUser.name ? thisUser.name : "未填写真实姓名" }}

</div>

<div class="prop-name">性别</div>

<el-radio-group v-model="thisUser.sex" class="prop-value" disabled>

<el-radio :label="1">男</el-radio>

<el-radio :label="0">女</el-radio>

</el-radio-group>

<div class="prop-name">年龄:</div>

<div class="prop-special-value">

{{ thisUser.age ? thisUser.age : "未填写年龄" }}

</div>

<div class="prop-name">手机号</div>

<div class="prop-value">

{{ thisUser.phone ? thisUser.phone : "未填写手机号" }}

</div>

<div class="prop-name">邮箱</div>

<div class="prop-value">

{{ thisUser.email ? thisUser.email : "未填写邮箱" }}

</div>

</div>

</template>

<script>

export default {

props: ["user"],

data() {

return {

thisUser: this.user,

};

},

created() {},

methods: {},

watch: {

user() {

this.thisUser = this.user;

},

},

};

</script>

<style scoped>

.prop-name {

margin-top: 15px;

margin-bottom: 6px;

font-weight: bold;

font-size: 20px;

font-family: "等线";

}

.prop-special-value {

margin-left: 10px;

font-family: Serif;

font-size: 18px;

color: #304683;

}

.prop-value {

font-size: 18px;

font-family: "等线";

}

</style>

### infocard.vue

<template>

<div class="container">

<h1>个人信息</h1>

<div class="title-liner"></div>

<el-row>

<el-col :span="16">

<my-info :user="user" />

</el-col>

<el-col :span="2"> </el-col>

<el-col :span="6">

<div class="icon">

<right-tab

@getInfo="getInfo"

@fatherChangeEditMode="fatherChangeEditMode"

:user="user"

/>

</div>

</el-col>

</el-row>

</div>

</template>

<script>

import RightTab from "./RightTab";

import MyInfo from "./Info";

export default {

props: ["user"],

data() {

return {

mode: 0, //0:展示信息的状态 1:编辑信息的状态

};

},

components: {

RightTab,

MyInfo,

},

created() {

this.getInfo();

},

activated() {

this.getInfo();

},

methods: {

fatherChangeEditMode(mode) {

this.mode = mode;

},

getInfo() {

this.$emit("getInfo");

},

},

watch: {

user() {

this.thisUser = this.user;

},

},

};

</script>

<style scoped>

.container {

padding: 30px;

background: #fff;

border: 1px solid #ddd;

border-radius: 5px;

width: 100%;

}

.title-liner {

border: 1px solid rgb(230, 230, 230);

background-color: rgb(230, 230, 230);

width: 100%;

}

.icon {

margin-top: 30px;

text-align: center;

}

</style>

### righttab.vue

<template>

<div>

<div class="icon-box">

<img

style="width:150px;height:150px;border-radius:50%;object-fit:fill;"

:src="user.icon"

onerror="onerror=null;src='https://img0.baidu.com/it/u=3730772664,138405132&fm=26&fmt=auto'"

/>

</div>

<br />

</div>

</template>

<script>

export default {

props: ["user"],

data() {

return {

thisUser: this.user,

mode: 0, //0:展示信息的状态 1:编辑信息的状态

changePassword: false,

param: {

oldPassword: "",

newPassword: "",

newPassword2: "",

},

tmpEducation: "",

};

},

created() {},

methods: {

changeEditMode() {

if (this.mode == 0) this.mode = 1;

else this.mode = 0;

this.$emit("fatherChangeEditMode", this.mode);

},

//编辑完成

async editFinished() {

if (this.mode == 0) this.mode = 1;

else this.mode = 0;

this.$emit("fatherChangeEditMode", this.mode);

await new Promise((resolve) => {

this.$http

.post("/editDoctorInfo/", JSON.stringify(this.thisUser))

.then((res) => {

if (res.data.success === false)

return this.$message.error(res.data.message);

this.$message.success(res.data.message);

});

resolve();

}).then(() => {});

},

//上传头像

onSuccess() {

this.$message.success({

title: "成功",

message: "上传成功！",

type: "success",

});

this.$emit("getInfo");

},

handlePreview(file) {

window.open(file.response.url);

},

//取消修改

async editCanceled() {

this.$emit("getInfo");

if (this.mode == 0) this.mode = 1;

else this.mode = 0;

this.$emit("fatherChangeEditMode", this.mode);

},

checkFile(file) {

var fileExtension = file.name.substring(file.name.lastIndexOf(".") + 1);

if (file.type.startsWith("image")) {

if (fileExtension === "gif") {

this.$message.error("不支持上传gif图片");

return false;

}

} else {

this.$message.error("请上传图片");

return false;

}

},

},

watch: {

user() {

this.thisUser = this.user;

},

},

};

</script>

<style scoped>

.edit-button {

width: 60%;

margin: 8px;

margin-bottom: 26px;

}

.edit-button-upload {

width: 100%;

/\* margin: 8px; \*/

}

.icon-box {

display: flex;

justify-content: center;

align-items: center;

}

.upload-demo {

width: 60%;

margin-top: 6px;

margin-bottom: 6px;

margin-left: 20%;

display: flex;

justify-content: center;

/\* margin: 8px; \*/

}

</style>

<style>

.el-upload {

width: 100%;

}

</style>

### middle.vue

<template>

<div class="container">

<div class="display">

<!-- 显示图片卡片 -->

<div v-loading="myProps.loadingNewPicture">

<div v-if="myProps.imageList.length > 0">

<div

v-for="singleImage in myProps.imageList"

:key="singleImage"

style="margin-top: 20px"

>

<photo-card

:singleImage="singleImage"

:props="myProps"

@getImageList="getImageList"

/>

</div>

<el-pagination

@size-change="handleSizeChange"

@current-change="handleCurrentChange"

:current-page="myProps.pagenum"

:page-sizes="[1, 3, 5, 10]"

:page-size="myProps.pagesize"

layout="total, sizes, prev, pager, next, jumper"

:total="myProps.total"

>

</el-pagination>

</div>

<div v-else>

<el-empty description="暂未上传图片" />

</div>

</div>

</div>

</div>

</template>

<script>

import PhotoCard from "@/components/Public/PhotoCard.vue";

export default {

components: {

PhotoCard,

},

data() {

return {

myProps: {

loadingNewPicture: false,

pagenum: 1,

pagesize: 5,

total: 0,

imageList: [],

},

};

},

created() {

this.getImageList();

},

methods: {

handleSizeChange(value) {

this.myProps.pagesize = value;

this.getImageList();

},

handleCurrentChange(value) {

this.myProps.pagenum = value;

this.getImageList();

},

async getImageList() {

this.myProps.loadingNewPicture = true;

await new Promise((resolve) => {

this.$http

.post(

"/getListForDoctor/",

JSON.stringify({

user\_id: this.$store.state.user\_id,

pagenum: this.myProps.pagenum,

pagesize: this.myProps.pagesize,

other\_user\_id: this.$route.query.id,

title:"",

showAll:false

})

)

.then((res) => {

{

if (res.data.message === "返回list成功") {

this.myProps.imageList = res.data.imageList;

this.myProps.total = res.data.total;

} else {

this.$message.error("获取列表失败");

}

this.myProps.loadingNewPicture = false;

}

});

resolve();

}).then(() => {});

},

},

};

</script>

<style scoped>

.container {

width: 100%;

text-align: center;

background-color: rgb(255, 255, 255, 0);

padding: 30px;

border: 0px solid #ddd;

border-radius: 5px;

}

.display {

width: 100%;

margin: 0 auto;

}

</style>

### Login/index.vue

<template>

<div class="login-wrap">

<div class="ms-login">

<div class="ms-title">眼底血管分割云平台</div>

<el-form

:model="param"

:rules="rules"

ref="login"

label-width="0px"

class="ms-content"

>

<el-form-item prop="username">

<el-input v-model="param.username" placeholder="username">

<template #prepend>

<i class="el-icon-user">用户名</i>

</template>

</el-input>

</el-form-item>

<el-form-item prop="password">

<el-input

type="password"

placeholder="password"

v-model="param.password"

@keyup.enter="submitForm()"

>

<template #prepend>

<i class="el-icon-lock">密码</i>

</template>

</el-input>

</el-form-item>

<div class="login-btn">

<el-button type="primary" @click="submitForm()">登录</el-button>

</div>

<h>当前还未注册？</h>

<h class="login-tips"

><router-link to="/register">前往注册</router-link></h

>

</el-form>

</div>

</div>

</template>

<script>

export default {

data() {

return {

param: {

username: "",

password: "",

},

rules: {

username: [

{ required: true, message: "请输入用户名", trigger: "blur" },

],

password: [

{ required: true, message: "请输入密码", trigger: "blur" },

],

},

};

},

created() {

this.$store.commit("clearTags");

},

methods: {

submitForm() {

this.$refs.login.validate(async (valid) => {

if (valid) {

await new Promise((resolve) => {

this.$http

.post("/login/", JSON.stringify(this.param))

.then((res) => {

if (res.data.success === false)

return this.$message.error(res.data.message);

this.$message.success("登录成功");

this.$store.commit("login", this.param.username);

this.$store.commit("storeId", res.data.userID);

console.log(res.data.userID);

this.$router.push({ path: "/doctor/upload" });

});

resolve();

}).then(() => {

console.log("成功啦post啦");

});

} else {

return;

}

});

},

},

};

</script>

<style scoped>

@import "style.css";

</style>

### Login/style.css

.login-wrap {

position: relative;

width: 100%;

height: 100%;

background-image: url(../../../assets/img/login.jpg);

}

.ms-title {

width: 100%;

line-height: 50px;

text-align: center;

font-size: 20px;

color: rgb(141, 139, 151);

border-bottom: 1px solid rgb(87, 107, 163);

}

.ms-login {

position: absolute;

left: 50%;

top: 50%;

width: 350px;

margin: -190px 0 0 -175px;

border-radius: 5px;

background: rgba(255, 255, 255);

overflow: hidden;

}

.ms-content {

padding: 30px 30px;

color: rgb(29, 29, 31);

}

.login-btn {

text-align: center;

}

.login-btn button {

width: 100%;

height: 36px;

margin-bottom: 10px;

}

.login-tips {

line-height: 30px;

color: rgb(29, 29, 31);

}

### Register/index.vue

<template>

<div class="register-wrap">

<el-scrollbar style="height: 100%">

<div class="ms-register">

<div class="ms-title">眼底血管分割云平台</div>

<el-form

:model="param"

:rules="rules"

ref="register"

class="demo-form-inline"

>

<el-form-item prop="username" label="用户名">

<el-input v-model="param.username" placeholder="请输入用户名">

</el-input>

</el-form-item>

<el-form-item prop="password" label="密码">

<el-input

type="password"

placeholder="请输入密码"

v-model="param.password"

@keyup.enter="submitForm()"

>

</el-input>

</el-form-item>

<el-form-item prop="password" label="确认密码">

<el-input

type="password"

placeholder="请确认密码"

v-model="param.tempPassword"

@keyup.enter="submitForm()"

>

</el-input>

</el-form-item>

<el-form-item prop="realname" label="真实姓名">

<el-input v-model="param.realname" placeholder="请输入真实姓名（选填）">

</el-input>

</el-form-item>

<el-form-item prop="email" label="邮箱">

<el-input v-model="param.email" placeholder="请输入邮箱（选填）">

</el-input>

</el-form-item>

<el-form-item prop="phone" label="手机号">

<el-input v-model="param.phone" placeholder="请输入手机号（选填）">

</el-input>

</el-form-item>

<el-form-item label="性别" prop="sex">

<el-radio-group v-model="param.sex" size="medium">

<el-radio border label="1">男</el-radio>

<el-radio border label="0">女</el-radio>

</el-radio-group>

</el-form-item>

<div class="register-btn">

<el-button type="primary" @click="submitForm()">注册</el-button>

</div>

<h>已有帐号？</h>

<h class="h">

<router-link to="/login">立即登录</router-link>

</h>

</el-form>

</div>

</el-scrollbar>

</div>

</template>

<script>

export default {

data() {

var checkEmail = (rule, value, callback) => {

const regEmail =

/(^([a-zA-Z0-9\_-])+@([a-zA-Z0-9\_-])+((.[a-zA-Z0-9\_-]{2,3}){1,2})+$)?/;

if (regEmail.test(value)) {

return callback();

}

callback(new Error("请输入合法的邮箱"));

};

var checkUsername = (rule, value, callback) => {

const regUsername = /(^[0-9]{11}$)\*/;

if (regUsername.test(value)) {

return callback();

}

callback(new Error("请输入合法的用户名"));

};

var checkPhone = (rule, value, callback) => {

const regPhone = /(^[0-9]+$)\*/;

if (regPhone.test(value)) {

return callback();

}

callback(new Error("请输入合法的手机号"));

};

return {

param: {

username: "",

password: "",

email: "",

phone: "",

sex: "",

},

rules: {

username: [

{ required: true, message: "请输入用户名", trigger: "blur" },

{

min: 1,

max: 20,

validator: checkUsername,

message: "用户名长度在1-20之间",

trigger: "blur",

},

],

password: [

{ required: true, message: "请输入密码", trigger: "blur" },

{

min: 6,

max: 15,

message: "密码长度在6到15个字符",

trigger: "blur",

},

],

email: [

{

validator: checkEmail,

trigger: "blur",

},

],

phone: [

{

validator: checkPhone,

trigger: "blur",

},

],

sex: [{ requierd: false, message: "请选择性别", trigger: "blur" }],

},

};

},

created() {

this.$store.commit("clearTags");

},

methods: {

submitForm() {

this.$refs.register.validate(async (valid) => {

if (valid) {

if (this.param.password !== this.param.tempPassword) {

return this.$message.error("输入的两次密码不一致");

}else {

await new Promise((resolve) => {

this.$http

.post("/register/", JSON.stringify(this.param))

.then((res) => {

console.log(res);

if (res.data.success === false)

return this.$message.error(res.data.message);

this.$message.success("注册成功");

this.$router.push({ path: "/login" });

});

resolve();

}).then(() => {

console.log("成功啦post啦");

});

}

} else {

return;

}

});

},

},

};

</script>

<style scoped>

@import "style.css";

</style>

### Register/style.css

.register-wrap {

position: relative;

width: 100%;

height: 100%;

background-image: url(../../../assets/img/login.jpg);

}

.ms-title {

width: 100%;

line-height: 50px;

text-align: center;

font-size: 20px;

color: rgb(141, 139, 151);

border-bottom: 1px solid rgb(87, 107, 163);

}

.ms-register {

margin: 0 auto;

width: 50%;

height: 50%;

border-radius: 5px;

background: rgba(255, 255, 255);

padding: 30px;

}

.register-btn {

text-align: center;

}

.register-btn button {

width: 100%;

height: 36px;

margin-bottom: 10px;

}

.register-tips {

font-size: 12px;

line-height: 30px;

color: rgb(29, 29, 31);

}

.h {

color: rgb(129, 109, 83);

}

### Menu.vue

<template>

<div class="left-menu">

<div class="icon-container">

<img

style="width:50px;height:50px;border-radius:50%;object-fit:fill;flex-shrink:0;"

:src="thisUser.icon"

onerror="onerror=null;src='https://img0.baidu.com/it/u=3730772664,138405132&fm=26&fmt=auto'"

/>

<span

class="user-name"

style="text-overflow:ellipsis; overflow:hidden;"

:title="thisUser.username"

>{{thisUser.username}}</span>

</div>

<div class="menu-container">

<el-menu

default-active="0"

class="el-menu-vertical-demo"

@select="handleSelectMenu"

>

<el-menu-item index="0">

<i class="el-icon-document"></i>

<span class="menu-text">个人信息</span>

</el-menu-item>

<el-menu-item index="1">

<i class="el-icon-menu"></i>

<span class="menu-text">图像列表</span>

</el-menu-item>

</el-menu>

</div>

</div>

</template>

<script>

export default {

props: ['user'],

data() {

return {

thisUser: this.user,

};

},

created() {

},

methods: {

handleSelectMenu(index) {

this.$emit("changeSelectedMenu", index)

}

},

watch: {

user() {

this.thisUser = this.user

}

}

};

</script>

<style scoped>

.menu-title {

display: flex;

justify-content: center;

align-items: center;

font-weight: 800;

font-size: 25px;

margin-bottom: 10px;

}

.user-name {

font-size: 25px;

font-weight: 800;

margin-left: 10px;

}

.icon-container {

display: flex;

align-items: center;

margin-bottom: 20px;

}

.left-menu{

width:240px;

margin: 0px;

}

</style>

### Infoedit.vue

<template>

<div>

<div class="prop-name">姓名</div>

<el-input v-model="thisUser.name" class="prop-value" />

<div class="prop-name">性别</div>

<el-radio-group v-model="thisUser.sex" class="prop-value">

<el-radio :label="1">男</el-radio>

<el-radio :label="0">女</el-radio>

</el-radio-group>

<div class="prop-name">身高:</div>

<el-input-number

v-model="thisUser.height"

class="prop-special-value"

:min="1"

:max="300"

/>

<div class="prop-name">体重:</div>

<el-input-number

v-model="thisUser.weight"

class="prop-special-value"

:min="1"

:max="500"

/>

<div class="prop-name">年龄:</div>

<el-input-number

v-model="thisUser.age"

class="prop-special-value"

:min="1"

:max="150"

/>

<div class="prop-name">手机号</div>

<el-input v-model="thisUser.phone" class="prop-value" />

<div class="prop-name">邮箱</div>

<el-input v-model="thisUser.email" class="prop-value" />

<div class="prop-name">地址</div>

<el-input v-model="thisUser.address" class="prop-value" />

</div>

</template>

<script>

export default {

props: ["user"],

data() {

return {

thisUser: this.user,

};

},

created() {},

methods: {},

watch: {

user() {

this.thisUser = this.user;

},

},

};

</script>

<style scoped>

.prop-name {

margin-top: 15px;

margin-bottom: 6px;

font-weight: bold;

font-size: 20px;

font-family: "等线";

}

.prop-special-value {

margin-left: 10px;

font-family: Serif;

font-size: 18px;

color: #304683;

}

.prop-value {

font-size: 18px;

font-family: "等线";

}

</style>