from django.contrib.auth import login

from django.contrib.auth.decorators import login\_required

from django.contrib.auth.models import User

from django.contrib.sites.shortcuts import get\_current\_site

from django.shortcuts import render, redirect

from django.utils.encoding import force\_bytes, force\_text

from django.utils.http import urlsafe\_base64\_encode, urlsafe\_base64\_decode

from django.template.loader import render\_to\_string

from .forms import SignUpForm

from .tokens import account\_activation\_token

@login\_required

def home(request):

return render(request, 'home.html')

#.decode()

def signup(request):

if request.method == 'POST':

form = SignUpForm(request.POST)

if form.is\_valid():

user = form.save(commit=False)

user.is\_active = False

user.save()

current\_site = get\_current\_site(request)

subject = 'Activate Your MySite Account'

message = render\_to\_string('account\_activation\_email.html', {

'user': user,

'domain': current\_site.domain,

'uid': urlsafe\_base64\_encode(force\_bytes(user.pk)),

'token': account\_activation\_token.make\_token(user),

})

user.email\_user(subject, message)

return redirect('account\_activation\_sent')

else:

form = SignUpForm()

return render(request, 'signup.html', {'form': form})

def account\_activation\_sent(request):

return render(request, 'account\_activation\_sent.html')

def activate(request, uidb64, token):

try:

uid = force\_text(urlsafe\_base64\_decode(uidb64))

user = User.objects.get(pk=uid)

except (TypeError, ValueError, OverflowError, User.DoesNotExist):

user = None

if user is not None and account\_activation\_token.check\_token(user, token):

user.is\_active = True

user.profile.email\_confirmed = True

user.save()

login(request, user)

return redirect('home')

else:

return render(request, 'account\_activation\_invalid.html')

from django.shortcuts import render,get\_object\_or\_404

from .forms import RequestForm

from .models import Request

from django.utils import timezone

from django.core.mail import BadHeaderError, send\_mail

from django.http import HttpResponse, HttpResponseRedirect

from django.shortcuts import redirect

from django.contrib.auth.decorators import login\_required

from django.conf import settings

# Create your views here.

def about(request):

return render(request, 'leave/about.html',{})

def contact(request):

return render(request, 'leave/contact.html',{})

def request\_list(request):

# print(Request.objects.all)

# requests = Request.objects.filter(created\_date\_\_lte=timezone.now()).order\_by('created\_date')

return render(request, 'leave/request\_list.html', {})

def all\_requests(request):

print(Request.objects.all)

# import pdb; pdb.set\_trace()

requests = Request.objects.filter(created\_date\_\_lte=timezone.now()).order\_by('created\_date')

return render(request, 'leave/list.html',{'requests':requests})

@login\_required

def request\_new(request):

if request.method == "POST":

# import pdb; pdb.set\_trace()

form = RequestForm(request.POST)

if form.is\_valid():

requests = form.save(commit=False)

requests.student = request.user

requests.created\_date = timezone.now()

# send\_mail(requests.subject, requests.description, settings.EMAIL\_HOST\_USER,

# [requests.parent\_email], fail\_silently=False)

requests.save()

return redirect('request\_detail', pk=requests.pk)

else:

form = RequestForm()

return render(request, 'leave/request\_edit.html', {'form': form})

def request\_detail(request, pk):

requests = get\_object\_or\_404(Request, pk=pk)

return render(request, 'leave/request\_detail.html', {'requests': requests})

@login\_required

def request\_edit(request, pk):

requests = get\_object\_or\_404(Request, pk=pk)

if request.method == "POST":

form = RequestForm(request.POST, instance=requests)

if form.is\_valid():

requests = form.save(commit=False)

requests.student = request.user

requests.created\_date = timezone.now()

requests.save()

return redirect('request\_detail', pk=requests.pk)

else:

form = RequestForm(instance=requests)

return render(request, 'leave/request\_edit.html', {'form': form})

def confirm(request, pk):

requests = get\_object\_or\_404(Request, pk=pk)

if requests.perm:

send\_mail(requests.subject, requests.description, settings.EMAIL\_HOST\_USER,

[requests.roll+'@iiita.ac.in'], fail\_silently=False)

'''def send\_email(request):

subject = request.POST.get('subject', '')

message = request.POST.get('message', '')

to\_email = request.POST.get('parent\_email', '')

if subject and message and from\_email:

try:

send\_mail(subject, message, 'abc@abc.com', ['admin@example.com'])

except BadHeaderError:

return HttpResponse('Invalid header found.')

return HttpResponseRedirect('/contact/thanks/')

else:

# In reality we'd use a form class

# to get proper validation errors.

return HttpResponse('Make sure all fields are entered and valid.')'''