from django.shortcuts import render, redirect

from .models import blogapp

from .forms import blogappCreate

from django.http import HttpResponse

#DataFlair

def index(request):

shelf = blogapp.objects.all()

return render(request, 'blogapp/library.html', {'shelf': shelf})

def upload(request):

upload = blogappCreate()

if request.method == 'POST':

upload = bloagappCreate(request.POST, request.FILES)

if upload.is\_valid():

upload.save()

return redirect('index')

else:

return HttpResponse("""your form is wrong, reload on <a href = "{{ url : 'index'}}">reload</a>""")

else:

return render(request, 'blogapp/upload\_form.html', {'upload\_form': upload})

def update\_book(request, blog\_id):

blog\_id = int(blog\_id)

try:

blog\_sell = blogapp.objects.get(id=blog\_id)

except blogapp.DoesNotExist:

return redirect('index')

blog\_form = BlogappCreate(request.POST or None, instance=blog\_sell)

if blog\_form.is\_valid():

blog\_form.save()

return redirect('index')

return render(request, 'blogapp/upload\_form.html', {'upload\_form': blog\_form})

def delete\_blogapp(request, blog\_id):

blog\_id = int(blog\_id)

try:

blog\_sell = blogapp.objects.get(id=blog\_id)

except Blogapp.DoesNotExist:

return redirect('index')

blog\_sell.delete()

return redirect('index')

class SearchResultsView(ListView):

model = blogapp

template\_name = 'search\_results.html'

queryset = blogapp.objects.filter(name\_\_icontains='Boston')

class RelatedModel(models.Model):

"""

A model with a foreign key.

"""

related = models.ForeignKey(to='self', delete\_blogapp=models.CASCADE)

history = AuditlogHistoryField()

class ManyRelatedModel(models.Model):

"""

A model with a many to many relation.

"""

related = models.ManyToManyField('self')

history = AuditlogHistoryField()