from django.shortcuts import render, get_object_or_404, redirect from django.http import JsonResponse from django.contrib import messages from django.urls import reverse from django.db.models import Q from django.utils import timezone

from .models import Universidad, Examen from .forms import UniversidadAdmisionForm, SolucionarioForm, ExamenForm from Login.models import UsuarioRol

def tiene_permiso_profesor_sobre(usuario, universidad): return (UsuarioRol.objects.filter(usuario=usuario, rol__nombre_rol='profesor').exists() and (usuario == universidad.profesor_creador or (usuario.codigo_modular == universidad.codigo_modular and usuario.institucion_educativa == universidad.institucion_educativa)))

def vista_repositorio(request): queryset = Universidad.objects.all().select_related('profesor_creador')

```
if request.user.is_authenticated and getattr(request.user, 'rol', None) ==
'profesor':
    queryset = queryset.filter(
        Q(codigo_modular=request.user.codigo_modular),
        Q(institucion_educativa=request.user.institucion_educativa)
    ).order_by('-fecha_creacion')

institucion_filter = request.GET.get('institucion')
tipo_filter = request.GET.get('tipo_solucionario')

if institucion_filter:
    queryset =
queryset.filter(institucion_educativa__iexact=institucion_filter)
if tipo_filter:
    queryset = queryset.filter(tipo_solucionario=tipo_filter)

context = {
    'universidades': queryset,
    'instituciones':
Universidad.objects.exclude(institucion_educativa__isnull=True)
                        .values_list('institucion_educativa', flat=True)
                        .distinct(),
    'tipos_solucionario': Universidad.TIPO_SOLUCIONARIO_CHOICES
}
return render(request, 'repositorio/repositorio.html', context)
```

def add_universidad(request): if not request.user.is_authenticated or not UsuarioRol.objects.filter(usuario=request.user, rol__nombre_rol='profesor').exists(): return redirect('repositorio')

```
tipo = request.POST.get('tipo_solucionario') or request.GET.get('tipo')

if tipo == 'admision':
    FormClass = UniversidadAdmisionForm
```

```
elif tipo in ['ejercicios', 'otro']:
    FormClass = SolucionarioForm
else:
    return render(request, 'repositorio/selector_tipo.html')

if request.method == 'POST':
    form = FormClass(request.POST, request.FILES, user=request.user)
    if form.is_valid():
        universidad = form.save(commit=False)
        universidad.profesor_creador = request.user
        universidad.codigo_modular = request.user.codigo_modular
        universidad.institucion_educativa =
request.user.institucion_educativa
        universidad.departamento = request.user.departamento
        universidad.provincia = request.user.provincia
        universidad.distrito = request.user.distrito
        universidad.save()

        if request.headers.get('X-Requested-With') == 'XMLHttpRequest':
            return JsonResponse({'success': True, 'redirect': True})

        messages.success(request, "Repositorio creado correctamente.")
        return redirect('repositorio')
    else:
        if request.headers.get('X-Requested-With') == 'XMLHttpRequest':
            return JsonResponse({'success': False, 'errors': form.errors},
status=400)
else:
    form = FormClass(user=request.user)

return render(request, 'repositorio/add_universidad.html', {'form': form,
'tipo': tipo})
```

def vista_examenes_universidad(request, universidad_id): return redirect('repositorio')

def editar_universidad(request, universidad_id): universidad = get_object_or_404(Universidad, id=universidad_id)

```
if not tiene_permiso_profesor_sobre(request.user, universidad):
    if request.headers.get('X-Requested-With') == 'XMLHttpRequest':
        return JsonResponse({'error': 'No tienes permisos para editar'},
status=403)
    messages.error(request, 'No tienes permisos para editar esta
universidad.')
    return redirect('repositorio')

if request.method == 'POST':
    form = UniversidadAdmisionForm(request.POST, request.FILES,
instance=universidad, user=request.user)
    if form.is_valid():
```

```
        universidad = form.save()

        if request.headers.get('X-Requested-With') == 'XMLHttpRequest':
            return JsonResponse({'success': True, 'redirect': True})

        messages.success(request, 'Repositorio actualizado correctamente.')
        return redirect('repositorio')
    else:
        if request.headers.get('X-Requested-With') == 'XMLHttpRequest':
            return JsonResponse({'success': False, 'errors': form.errors},
status=400)
else:
    form = UniversidadAdmisionForm(instance=universidad, user=request.user)

return render(request, 'repositorio/add_universidad.html', {
    'form': form,
    'universidad': universidad,
    'editing': True
})
```

def examenes_ajax(request, universidad_id): universidad = get_object_or_404(Universidad, id=universidad_id) examenes = universidad.examenes.all()

```
examenes_data = [
    {
        'id': examen.id,
        'nombre': examen.nombre,
        'fecha': str(examen.fecha),
        'archivo': examen.archivo.url if examen.archivo else ''
    }
    for examen in examenes
]

return JsonResponse({'examenes': examenes_data})
```

def add_examen(request, universidad_id): universidad = get_object_or_404(Universidad, id=universidad_id)

```
if not tiene_permiso_profesor_sobre(request.user, universidad):
    return JsonResponse({'error': 'Acceso no autorizado'}, status=403)

if request.method == 'POST':
    form = ExamenForm(request.POST, request.FILES)
    if form.is_valid():
        examen = form.save(commit=False)
        examen.universidad = universidad
        examen.codigo_modular = universidad.codigo_modular
        examen.save()
```

```
        if request.headers.get('x-requested-with') == 'XMLHttpRequest':
            return JsonResponse({
                'success': True,
                'message': 'Examen agregado exitosamente',
                'examen_id': examen.id
            })

        messages.success(request, 'Examen agregado correctamente')
        return redirect('vista_examenes_universidad',
universidad_id=universidad.id)
    else:
        if request.headers.get('x-requested-with') == 'XMLHttpRequest':
            return JsonResponse({'success': False, 'errors': form.errors},
status=400)
else:
    form = ExamenForm(initial={'fecha': timezone.now().strftime('%Y-%m-%d')})

return render(request, 'repositorio/examenes_por_universidad.html', {
    'universidad': universidad,
    'form': form,
    'mostrar_formulario': True
})
```

def eliminar_universidad(request, universidad_id): universidad = get_object_or_404(Universidad, id=universidad_id)

```
if not tiene_permiso_profesor_sobre(request.user, universidad):
    return JsonResponse({
        'success': False,
        'error': 'No tienes permisos para eliminar este examen'
    }, status=403)

try:
    universidad.delete()
    return JsonResponse({
        'success': True,
        'message': 'Universidad eliminada correctamente',
        'redirect_url': reverse('repositorio')
    })
except Exception as e:
    return JsonResponse({'success': False, 'error': str(e)}, status=500)
```

def editar_examen(request, examen_id): examen = get_object_or_404(Examen, id=examen_id) form = ExamenForm(request.POST or None, request.FILES or None, instance=examen)

```
if request.method == 'POST' and form.is_valid():
    form.save()
    messages.success(request, 'Examen actualizado correctamente.')
    return redirect('repositorio')
```

```
if request.headers.get('x-requested-with') == 'XMLHttpRequest':
    return render(request, 'repositorio/form_fragmento.html', {'form': form})

return JsonResponse({'error': 'Acceso no permitido sin AJAX'}, status=400)
```

def eliminar_examen(request, examen_id): examen = get_object_or_404(Examen, id=examen_id) universidad = examen.universidad

```
if not tiene_permiso_profesor_sobre(request.user, universidad):
    return JsonResponse({
        'success': False,
        'error': 'No tienes permisos para eliminar este examen'
    }, status=403)

try:
    examen.delete()
    return JsonResponse({'success': True, 'message': 'Examen eliminado
correctamente'})
except Exception as e:
    return JsonResponse({'success': False, 'error': str(e)}, status=500)
```