Le plan de recrutement". Cette fonctionnalité permettra de gérer et de suivre les plans de recrutement de l'entreprise.

1. Modèles (ajoutez ceci à models.py) :

from django.db import models

from django.utils import timezone

class PlanRecrutement(models.Model):

STATUT\_CHOICES = [

('EN\_COURS', 'En cours'),

('TERMINE', 'Terminé'),

('ANNULE', 'Annulé'),

]

titre = models.CharField(max\_length=200)

description = models.TextField()

date\_debut = models.DateField()

date\_fin = models.DateField()

nombre\_postes = models.IntegerField()

competences\_requises = models.TextField()

departement = models.CharField(max\_length=100)

statut = models.CharField(max\_length=20, choices=STATUT\_CHOICES, default='EN\_COURS')

def \_\_str\_\_(self):

return self.titre

class CandidatPlan(models.Model):

STATUT\_CHOICES = [

('RECU', 'Reçu'),

('EN\_COURS', 'En cours'),

('ACCEPTE', 'Accepté'),

('REFUSE', 'Refusé'),

]

plan = models.ForeignKey(PlanRecrutement, on\_delete=models.CASCADE, related\_name='candidats')

nom = models.CharField(max\_length=100)

prenom = models.CharField(max\_length=100)

email = models.EmailField()

telephone = models.CharField(max\_length=20)

cv = models.FileField(upload\_to='cvs/')

date\_candidature = models.DateField(default=timezone.now)

statut = models.CharField(max\_length=20, choices=STATUT\_CHOICES, default='RECU')

commentaires = models.TextField(blank=True)

def \_\_str\_\_(self):

return f"{self.nom} {self.prenom} - {self.plan.titre}"

1. Formulaires (ajoutez ceci à forms.py) :

from django import forms

from .models import PlanRecrutement, CandidatPlan

class PlanRecrutementForm(forms.ModelForm):

class Meta:

model = PlanRecrutement

fields = ['titre', 'description', 'date\_debut', 'date\_fin', 'nombre\_postes', 'competences\_requises', 'departement', 'statut']

widgets = {

'date\_debut': forms.DateInput(attrs={'type': 'date'}),

'date\_fin': forms.DateInput(attrs={'type': 'date'}),

}

class CandidatPlanForm(forms.ModelForm):

class Meta:

model = CandidatPlan

fields = ['plan', 'nom', 'prenom', 'email', 'telephone', 'cv', 'statut', 'commentaires']

1. Vues (ajoutez ceci à views.py) :

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

from django.contrib import messages

from .models import PlanRecrutement, CandidatPlan

from .forms import PlanRecrutementForm, CandidatPlanForm

def liste\_plans\_recrutement(request):

plans = PlanRecrutement.objects.all().order\_by('-date\_debut')

return render(request, 'gpec/liste\_plans\_recrutement.html', {'plans': plans})

def ajouter\_plan\_recrutement(request):

if request.method == 'POST':

form = PlanRecrutementForm(request.POST)

if form.is\_valid():

form.save()

messages.success(request, "Plan de recrutement ajouté avec succès.")

return redirect('liste\_plans\_recrutement')

else:

form = PlanRecrutementForm()

return render(request, 'gpec/ajouter\_plan\_recrutement.html', {'form': form})

def detail\_plan\_recrutement(request, plan\_id):

plan = get\_object\_or\_404(PlanRecrutement, id=plan\_id)

candidats = plan.candidats.all().order\_by('-date\_candidature')

return render(request, 'gpec/detail\_plan\_recrutement.html', {'plan': plan, 'candidats': candidats})

def ajouter\_candidat(request, plan\_id):

plan = get\_object\_or\_404(PlanRecrutement, id=plan\_id)

if request.method == 'POST':

form = CandidatPlanForm(request.POST, request.FILES)

if form.is\_valid():

candidat = form.save(commit=False)

candidat.plan = plan

candidat.save()

messages.success(request, "Candidat ajouté avec succès.")

return redirect('detail\_plan\_recrutement', plan\_id=plan.id)

else:

form = CandidatPlanForm(initial={'plan': plan})

return render(request, 'gpec/ajouter\_candidat.html', {'form': form, 'plan': plan})

def modifier\_statut\_candidat(request, candidat\_id):

candidat = get\_object\_or\_404(CandidatPlan, id=candidat\_id)

if request.method == 'POST':

nouveau\_statut = request.POST.get('statut')

if nouveau\_statut in dict(CandidatPlan.STATUT\_CHOICES):

candidat.statut = nouveau\_statut

candidat.save()

messages.success(request, "Statut du candidat mis à jour.")

else:

messages.error(request, "Statut invalide.")

return redirect('detail\_plan\_recrutement', plan\_id=candidat.plan.id)

1. URLs (ajoutez ceci à urls.py) :

from django.urls import path

from . import views

urlpatterns = [

# ... autres URLs ...

path('gpec/plans-recrutement/', views.liste\_plans\_recrutement, name='liste\_plans\_recrutement'),

path('gpec/ajouter-plan-recrutement/', views.ajouter\_plan\_recrutement, name='ajouter\_plan\_recrutement'),

path('gpec/plan-recrutement/<int:plan\_id>/', views.detail\_plan\_recrutement, name='detail\_plan\_recrutement'),

path('gpec/ajouter-candidat/<int:plan\_id>/', views.ajouter\_candidat, name='ajouter\_candidat'),

path('gpec/modifier-statut-candidat/<int:candidat\_id>/', views.modifier\_statut\_candidat, name='modifier\_statut\_candidat'),

]

1. Templates :
   1. liste\_plans\_recrutement.html :

{% extends 'base.html' %}

{% block content %}

<h1>Plans de Recrutement</h1>

<a href="{% url 'ajouter\_plan\_recrutement' %}" class="btn btn-primary mb-3">Ajouter un plan de recrutement</a>

<table class="table">

<thead>

<tr>

<th>Titre</th>

<th>Date de début</th>

<th>Date de fin</th>

<th>Nombre de postes</th>

<th>Département</th>

<th>Statut</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

{% for plan in plans %}

<tr>

<td>{{ plan.titre }}</td>

<td>{{ plan.date\_debut }}</td>

<td>{{ plan.date\_fin }}</td>

<td>{{ plan.nombre\_postes }}</td>

<td>{{ plan.departement }}</td>

<td>{{ plan.get\_statut\_display }}</td>

<td>

<a href="{% url 'detail\_plan\_recrutement' plan.id %}" class="btn btn-info btn-sm">Détails</a>

</td>

</tr>

{% endfor %}

</tbody>

</table>

{% endblock %}

* 1. ajouter\_plan\_recrutement.html :

{% extends 'base.html' %}

{% block content %}

<h1>Ajouter un Plan de Recrutement</h1>

<form method="post">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit" class="btn btn-primary">Enregistrer</button>

</form>

{% endblock %}

* 1. detail\_plan\_recrutement.html :

{% extends 'base.html' %}

{% block content %}

<h1>{{ plan.titre }}</h1>

<div class="card mb-3">

<div class="card-body">

<h5 class="card-title">Détails du plan</h5>

<p><strong>Description :</strong> {{ plan.description }}</p>

<p><strong>Date de début :</strong> {{ plan.date\_debut }}</p>

<p><strong>Date de fin :</strong> {{ plan.date\_fin }}</p>

<p><strong>Nombre de postes :</strong> {{ plan.nombre\_postes }}</p>

<p><strong>Compétences requises :</strong> {{ plan.competences\_requises }}</p>

<p><strong>Département :</strong> {{ plan.departement }}</p>

<p><strong>Statut :</strong> {{ plan.get\_statut\_display }}</p>

</div>

</div>

<h2>Candidats</h2>

<a href="{% url 'ajouter\_candidat' plan.id %}" class="btn btn-primary mb-3">Ajouter un candidat</a>

<table class="table">

<thead>

<tr>

<th>Nom</th>

<th>Prénom</th>

<th>Email</th>

<th>Date de candidature</th>

<th>Statut</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

{% for candidat in candidats %}

<tr>

<td>{{ candidat.nom }}</td>

<td>{{ candidat.prenom }}</td>

<td>{{ candidat.email }}</td>

<td>{{ candidat.date\_candidature }}</td>

<td>

<form method="post" action="{% url 'modifier\_statut\_candidat' candidat.id %}">

{% csrf\_token %}

<select name="statut" onchange="this.form.submit()">

{% for value, label in candidat.STATUT\_CHOICES %}

<option value="{{ value }}" {% if candidat.statut == value %}selected{% endif %}>{{ label }}</option>

{% endfor %}

</select>

</form>

</td>

<td>

<a href="{{ candidat.cv.url }}" class="btn btn-secondary btn-sm" target="\_blank">Voir CV</a>

</td>

</tr>

{% endfor %}

</tbody>

</table>

{% endblock %}

* 1. ajouter\_candidat.html :

{% extends 'base.html' %}

{% block content %}

<h1>Ajouter un Candidat pour {{ plan.titre }}</h1>

<form method="post" enctype="multipart/form-data">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit" class="btn btn-primary">Enregistrer</button>

</form>

{% endblock %}

N'oubliez pas d'effectuer les migrations après avoir ajouté ces nouveaux modèles :

python manage.py makemigrations

python manage.py migrate

Cette implémentation vous permet de :

1. Créer et gérer des plans de recrutement
2. Ajouter des candidats à chaque plan de recrutement
3. Suivre le statut de chaque candidat
4. Voir les détails de chaque plan de recrutement et ses candidats associés

Assurez-vous d'ajouter un lien vers cette nouvelle fonctionnalité dans votre dashboard GPEC ou dans le menu de navigation principal.

Cette fonctionnalité vous donnera un outil complet pour gérer vos plans de recrutement, ce qui est essentiel pour une planification efficace des ressources humaines et pour suivre les processus de recrutement en cours.