implémenter l'outil d'évaluation :

1. Modèle d'évaluation (dans models.py) :

from django.db import models

from employe.models import Employe

class Evaluation(models.Model):

employe = models.ForeignKey(Employe, on\_delete=models.CASCADE)

evaluateur = models.ForeignKey(Employe, on\_delete=models.CASCADE, related\_name='evaluations\_effectuees')

date\_evaluation = models.DateField()

# Critères d'évaluation

performance\_poste = models.IntegerField(choices=[(i, i) for i in range(1, 6)])

qualites\_relationnelles = models.IntegerField(choices=[(i, i) for i in range(1, 6)])

engagement\_motivation = models.IntegerField(choices=[(i, i) for i in range(1, 6)])

autonomie = models.IntegerField(choices=[(i, i) for i in range(1, 6)])

conformite\_organisationnelle = models.IntegerField(choices=[(i, i) for i in range(1, 6)])

commentaires\_collaborateur = models.TextField(blank=True)

commentaires\_superieur = models.TextField(blank=True)

formations\_suivies = models.TextField(blank=True)

besoins\_formation = models.TextField(blank=True)

def moyenne\_globale(self):

return (self.performance\_poste + self.qualites\_relationnelles +

self.engagement\_motivation + self.autonomie +

self.conformite\_organisationnelle) / 5

1. Formulaire d'évaluation (créez un nouveau fichier forms.py) :

from django import forms

from .models import Evaluation

class EvaluationForm(forms.ModelForm):

class Meta:

model = Evaluation

fields = ['employe', 'evaluateur', 'date\_evaluation', 'performance\_poste',

'qualites\_relationnelles', 'engagement\_motivation', 'autonomie',

'conformite\_organisationnelle', 'commentaires\_collaborateur',

'commentaires\_superieur', 'formations\_suivies', 'besoins\_formation']

widgets = {

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

'commentaires\_collaborateur': forms.Textarea(attrs={'rows': 4}),

'commentaires\_superieur': forms.Textarea(attrs={'rows': 4}),

'formations\_suivies': forms.Textarea(attrs={'rows': 3}),

'besoins\_formation': forms.Textarea(attrs={'rows': 3}),

}

1. Vue pour l'évaluation (dans views.py) :

from django.shortcuts import render, redirect

from .models import Evaluation

from .forms import EvaluationForm

def evaluation(request):

if request.method == 'POST':

form = EvaluationForm(request.POST)

if form.is\_valid():

form.save()

return redirect('liste\_evaluations')

else:

form = EvaluationForm()

return render(request, 'gpec/evaluation\_form.html', {'form': form})

def liste\_evaluations(request):

evaluations = Evaluation.objects.all()

return render(request, 'gpec/liste\_evaluations.html', {'evaluations': evaluations})

1. Template pour le formulaire d'évaluation (evaluation\_form.html) :

{% extends 'base.html' %}

{% block content %}

<h1>Formulaire d'Évaluation</h1>

<form method="post">

{% csrf\_token %}

{{ form.as\_p }}

<button type="submit" class="btn btn-primary">Soumettre l'évaluation</button>

</form>

{% endblock %}

1. Template pour la liste des évaluations (liste\_evaluations.html) :

{% extends 'base.html' %}

{% block content %}

<h1>Liste des Évaluations</h1>

<table class="table">

<thead>

<tr>

<th>Employé</th>

<th>Date d'évaluation</th>

<th>Moyenne globale</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

{% for evaluation in evaluations %}

<tr>

<td>{{ evaluation.employe }}</td>

<td>{{ evaluation.date\_evaluation }}</td>

<td>{{ evaluation.moyenne\_globale|floatformat:2 }}</td>

<td>

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

</td>

</tr>

{% endfor %}

</tbody>

</table>

<a href="{% url 'evaluation' %}" class="btn btn-primary">Nouvelle évaluation</a>

{% endblock %}

1. Mise à jour des URLs (dans urls.py) :

from django.urls import path

from . import views

urlpatterns = [

# ... autres URLs ...

path('gpec/evaluation/', views.evaluation, name='evaluation'),

path('gpec/evaluations/', views.liste\_evaluations, name='liste\_evaluations'),

# Ajoutez une URL pour les détails de l'évaluation si nécessaire

]