

## TP4 Python Django – Les vues

fait par :

KOUTEMA Ditoma

## **Table des matières**

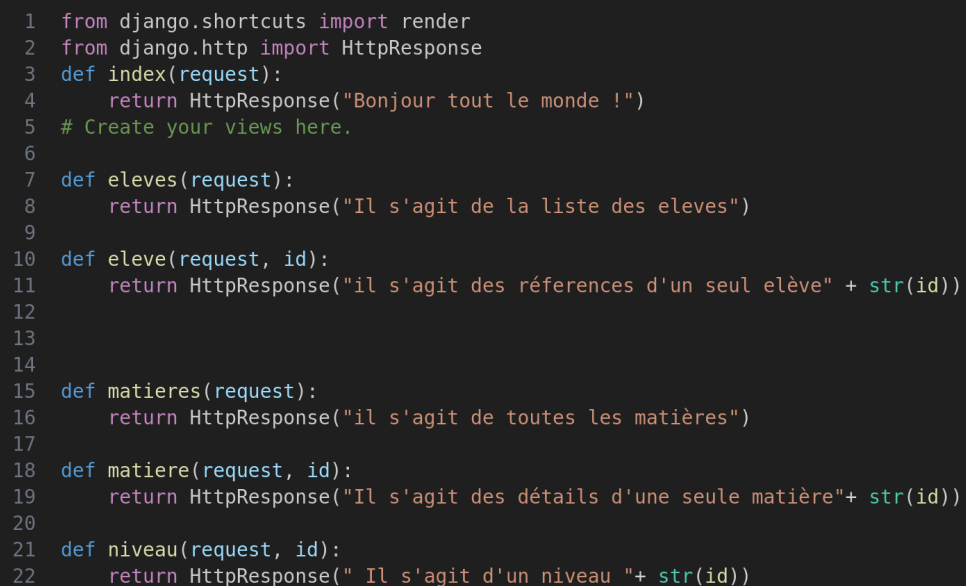
<b>1</b>	<b>Mise en route</b>	<b>3</b>
<b>2</b>	<b>Écriture de vues en code Python</b>	<b>3</b>
<b>3</b>	<b>Les templates</b>	<b>5</b>
<b>4</b>	<b>Gestion des erreurs</b>	<b>5</b>
<b>5</b>	<b>Navigation et espaces de noms</b>	<b>7</b>

## 1 Mise en route

- Une vue est une classe contenant les fonctions de contrôles de l'application.
- Nous allons travailler aujourd'hui sur le fichier [views.py](#)
- Pour lancer le serveur on fait `py manage.py runserver localhost :8000`

## 2 Écriture de vues en code Python

1. L'URL de la racine d'application est : <http://localhost:8000/notes/>
2. Il s'agit de l'id qu'on ajoute aux paramètres. L'id la permet d'avoir les détails d'un et un seul objet.

A screenshot of a code editor with a dark background and light-colored text. The code is written in Python and defines several Django view functions. The functions are: `index` (returns a 'Bonjour tout le monde !' message), `eleves` (returns 'Il s'agit de la liste des eleves'), `eleve` (returns a message with a student ID), `matieres` (returns 'il s'agit de toutes les matières'), `matiere` (returns a message with a subject ID), and `niveau` (returns a message with a level ID). The code is numbered from 1 to 22.

```
1 from django.shortcuts import render
2 from django.http import HttpResponse
3 def index(request):
4     return HttpResponse("Bonjour tout le monde !")
5     # Create your views here.
6
7 def eleves(request):
8     return HttpResponse("Il s'agit de la liste des eleves")
9
10 def eleve(request, id):
11     return HttpResponse("il s'agit des références d'un seul élève" + str(id))
12
13
14
15 def matieres(request):
16     return HttpResponse("il s'agit de toutes les matières")
17
18 def matiere(request, id):
19     return HttpResponse("Il s'agit des détails d'une seule matière" + str(id))
20
21 def niveau(request, id):
22     return HttpResponse(" Il s'agit d'un niveau " + str(id))
```

3. Pour que ces vues soient accessibles il faut que je définisse l'URI de chaque méthode contenant le chemin de la vue.
- Je vais le faire dans le fichier [urls.py](#)

```

1  from django.urls import path
2  from . import views
3  urlpatterns = [
4
5      path('', views.index, name='index'),
6
7      path("eleves/", views.eleves, name='eleves'),
8      path("eleve/<int:id>/", views.eleve, name='eleve'),
9
10     path("matieres/", views.matieres, name='matieres'),
11     path("matiere/<int:id>/", views.matiere, name='matiere'),
12
13     path("niveau/<int:id>/", views.niveau, name='niveau'),
14 ]
15

```

4. Pour accéder à la page de détails :  
 \_Je prend pour exemple pour élève donc j'ai fais :  
<http://localhost:8000/notes/eleve/2/>  
 \_Pour matière :  
<http://localhost:8000/notes/matiere/1/>
5. Les modèles qui doivent être appelés pour chacune des vues : modèles élève, matière, niveau.

```

1  from notes.models import Eleve, Matiere
2  def eleves(request):
3      lesEleves = Eleve.objects.all()
4      return HttpResponse(lesEleves)
5
6
7  def matieres(request):
8      lesmatiers = Matiere.objects.all()
9      #return HttpResponse("il s'agit de toutes les matières")
10     return HttpResponse(lesmatiers)
11
12

```

### 3 Les templates

1. Un template est un moteur de rendue de données.
2. Ce code ne vas rien faire.
3. La fonction `render` permet de retourner un template.
4. Report du code .

```
1 def eleves(request):
2     lesEleves = Eleve.objects.all()
3     return render(request, "eleves/index.html", {'lesEleves' : lesEleves})
4
5
6 def eleve(request, id):
7     return HttpResponse("il s'agit des références d'un seul élève" + str(id))
8
9
10
11 def matieres(request):
12     lesmatiers = Matiere.objects.all()
13     return render(request, "matieres/index.html", {'lesmatiers' : lesmatiers})
```

Contenu des fichier matiere/index.html et eleves/index.html

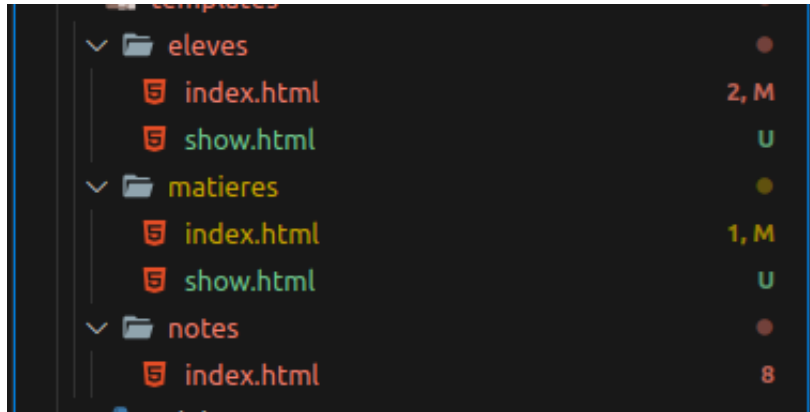
```
1 {% for metes in lesmatiers %}
2     <tr>
3         <td class="px-4 py-4 text-sm font-medium whitespace-nowrap">
4             <div>
5                 <h2 class="font-medium text-gray-800 dark:text-white ">{{ metes.nom }}</h2>
6             </div>
7         </td>
8     </tr>
9 {% endfor %}
```

### 4 Gestion des erreurs

1. Les autres vues sont différents parce qu'elles comportent des paramètre.
2. La fonction `get_object_or_404` appelle `get()` d'un gestionnaire de modèle donné, mais génère une exception `Http404` au lieu de l'exception `DoesNotExist` du modèle.

Cette fonction nous sera utile dans les autres vues pour la simple raison que si on retourne une vue qui n'existe pas alors on aura une erreur 404.

### 3. Architecture du templates



\_Contenu du fichier views.py

```
1 from django.shortcuts import render
2 from django.http import Http404, HttpResponse
3 from notes.models import Eleve, Matiere
4 def index(request):
5     return render(request, "notes/index.html")
6     # Create your views here.
7
8 def eleves(request):
9     lesEleves = Eleve.objects.all()
10    return render(request, "eleves/index.html", {'lesEleves': lesEleves})
11
12
13 def eleve(request, id):
14     try:
15         unEleve = Eleve.objects.get(id=id)
16         return render(request, "eleves/show.html", {'unEleve': unEleve})
17     except Eleve.DoesNotExist:
18         raise Http404("Model eleve n'existe pas ")
19
20
21
22 def matieres(request):
23     lesmatiers = Matiere.objects.all()
24     return render(request, "matieres/index.html", {'lesmatiers': lesmatiers})
25
26 def matiere(request, id):
27     try:
28         uneMatiere = Matiere.objects.get(id=id)
29         return render(request, "matieres/show.html", {'uneMatiere': uneMatiere})
30     except Matiere.DoesNotExist:
31         raise Http404("Model Matiere n'existe pas ")
32
33
34 def niveau(request, id):
35     return HttpResponse(" Il s'agit d'un niveau "+ str(id))
```

### 4. La ligne de code qu'on doit supprimer

```
from django.http import HttpResponse
```

## 5 Navigation et espaces de noms

### 1. Reporte de tous les templates

```
<table class="min-w-full divide-y ■divide-gray-200 □dark:divide-gray-700">
  <thead class="■bg-gray-50 □dark:bg-gray-800">
    <tr>
      <th scope="col" class="py-3.5 px-4 text-sm font-normal text-left rtl:text-right ■text-gray-500 ■dark:text-white">
        <button class="flex items-center gap-x-3 focus:outline-none">
          <span>Noms des matieres</span>
          <svg class="h-3" viewBox="0 0 10 11" fill="none" xmlns="http://www.w3.org/2000/svg">
            <path d="M2.13347 0.0999756H2.98516L5.01902 4.79058H3.86226L3.45549 3.79907H1.63772L1.01902 4.79058H0.0999756" stroke="■text-gray-500 □dark:text-white" stroke-width="1.5">
            <path d="M0.722656 9.60832L3.09974 6.78633H0.811638V5.87109H4.35819V6.78633L2.01925 9.60832" stroke="■text-gray-500 □dark:text-white" stroke-width="1.5">
            <path d="M8.45558 7.25664V7.40664H8.60558H9.66065C9.72481 7.40664 9.74667 7.42274 9.75168 7.44888" stroke="■text-gray-500 □dark:text-white" stroke-width="1.5">
          </svg>
        </button>
      </th>
      <th scope="col" class="px-4 py-3.5 text-sm font-normal text-left rtl:text-right ■text-gray-500 ■dark:text-white">
        Details matieres
      </th>
    </tr>
  </thead>
  <tbody class="■bg-white divide-y ■divide-gray-200 □dark:divide-gray-700 □dark:bg-gray-900">
    {% for metes in lesmatieres %}
    <tr>
      <td class="px-4 py-4 text-sm font-medium whitespace-nowrap">
        <div>
          <h2 class="font-medium □text-gray-800 ■dark:text-white ">{{ metes.nom }}</h2>
        </div>
      </td>
      <td class="px-4 py-4 text-sm whitespace-nowrap">
        <a href="{% url 'matiere' metes.id %}">
          <svg xmlns="http://www.w3.org/2000/svg" fill="none" viewBox="0 0 24 24" stroke-width="1.5" stroke="■text-gray-500 □dark:text-white">
            <path stroke-linecap="round" stroke-linejoin="round" d="M2.036 12.322a1.012 1.012 0 010 0.109" stroke="■text-gray-500 □dark:text-white">
            <path stroke-linecap="round" stroke-linejoin="round" d="M15 12a3 3 0 11-6 0 3 3 0 016 0" stroke="■text-gray-500 □dark:text-white">
          </svg>
        </a>
      </td>
    </tr>
    {% endfor %}
  </tbody>
</table>
```

### 2. Pour les améliorer nous devons ajouter du style pour une meilleur visibilité.