

# TP5 – Quelques vues de détail

## Contexte et Objectifs

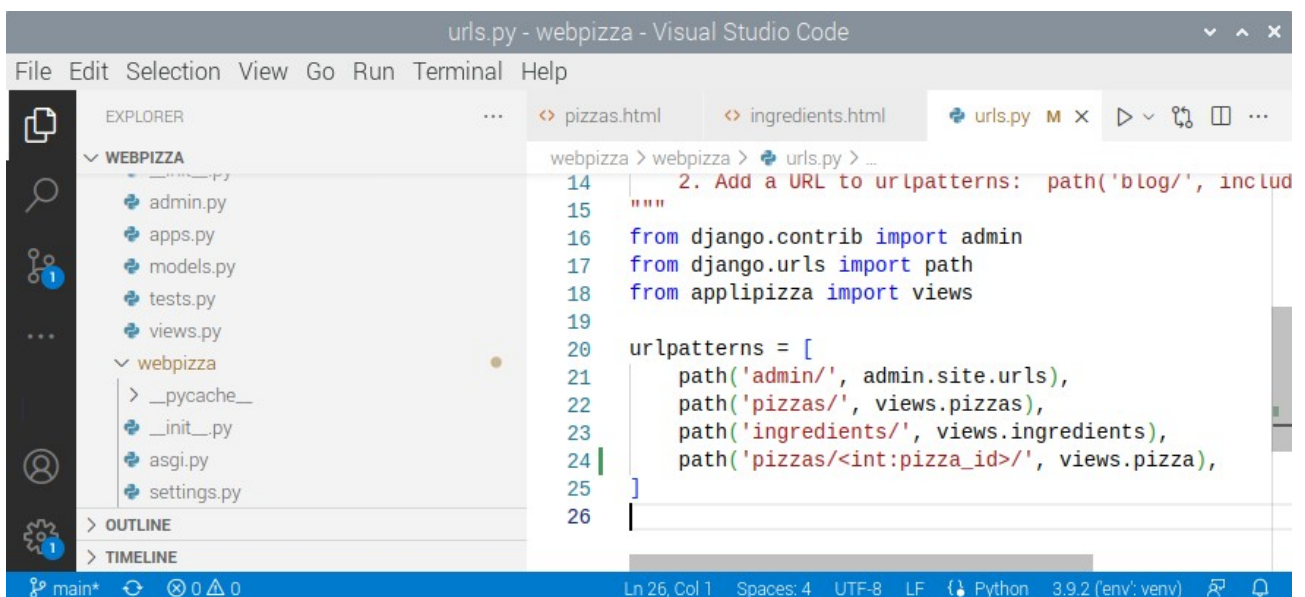
Les premières vues, améliorées par le gabarit base.html, font des listes des pizzas et d'ingrédients. Nous allons agrémenter les vues pizzas de nouvelles vues, celles qui donnent les détails sur une pizza. Nous mettrons aussi en place les url vers ces vues détail, ainsi que des liens vers ces vues dans la liste globale des pizzas.

## Modification du fichier urls.py

1. Nous allons créer une url générique pour le détail d'une pizza. Nous pourrions l'appeler ainsi :

<http://127.0.0.1:8000/pizzas/1/>  
<http://127.0.0.1:8000/pizzas/2/>

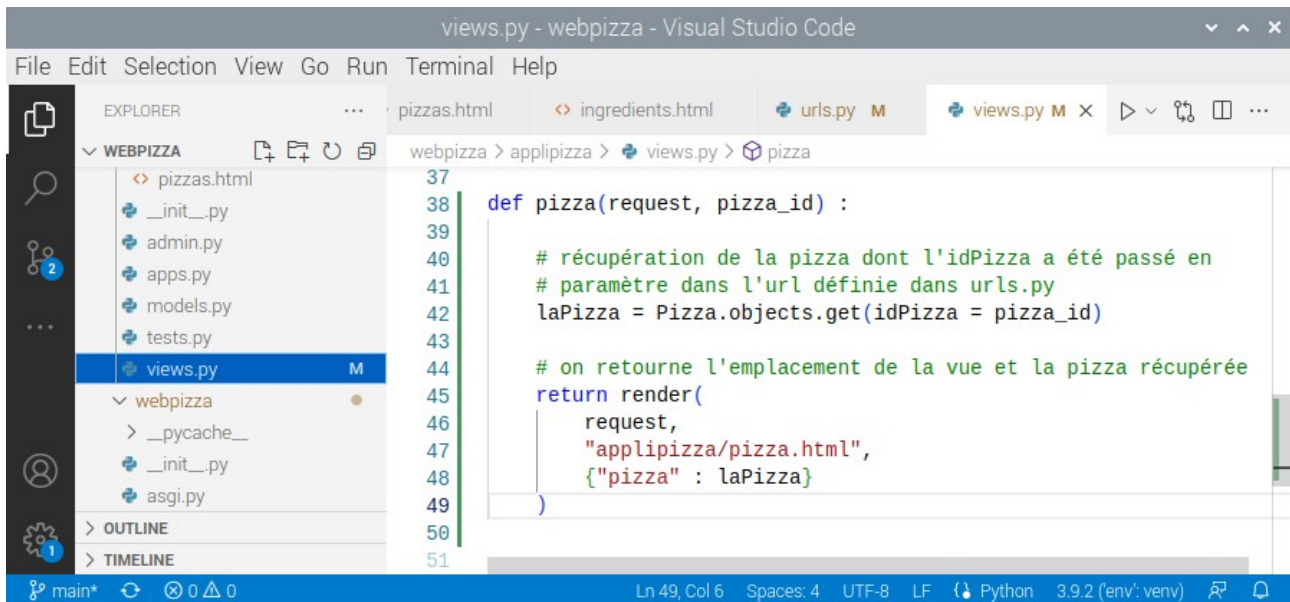
Dans le fichier urls.py, ajoutez le path de la dernière ligne (il contient un int variable nommé pizza\_id), path qui envoie vers une nouvelle vue « pizza », pas encore codée dans le fichier applipizza/views.py.



```
urls.py - webpizza - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
WEBPIZZA
  admin.py
  apps.py
  models.py
  tests.py
  views.py
  webpizza
    __pycache__
    __init__.py
    asgi.py
    settings.py
OUTLINE
TIMELINE
main* 0 0 0
Ln 26, Col 1 Spaces: 4 UTF-8 LF Python 3.9.2 (env: venv)
```

```
14 2. Add a URL to urlpatterns: path('blog/', includ
15 """
16 from django.contrib import admin
17 from django.urls import path
18 from applipizza import views
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('pizzas/', views.pizzas),
23     path('ingredients/', views.ingredients),
24     path('pizzas/<int:pizza_id>/', views.pizza),
25 ]
26
```

2. Créez dans `applipizza/views.py` une vue `pizza` dont le code est le suivant. Vous remarquerez qu'elle gère un paramètre supplémentaire `pizza_id` qui correspond à la variable du path.



```
views.py - webpizza - Visual Studio Code
File Edit Selection View Go Run Terminal Help

EXPLORER
  WEBPIZZA
    pizzas.html
    __init__.py
    admin.py
    apps.py
    models.py
    tests.py
    views.py M
  webpizza
    __pycache__
    __init__.py
    asgi.py
  OUTLINE
  TIMELINE

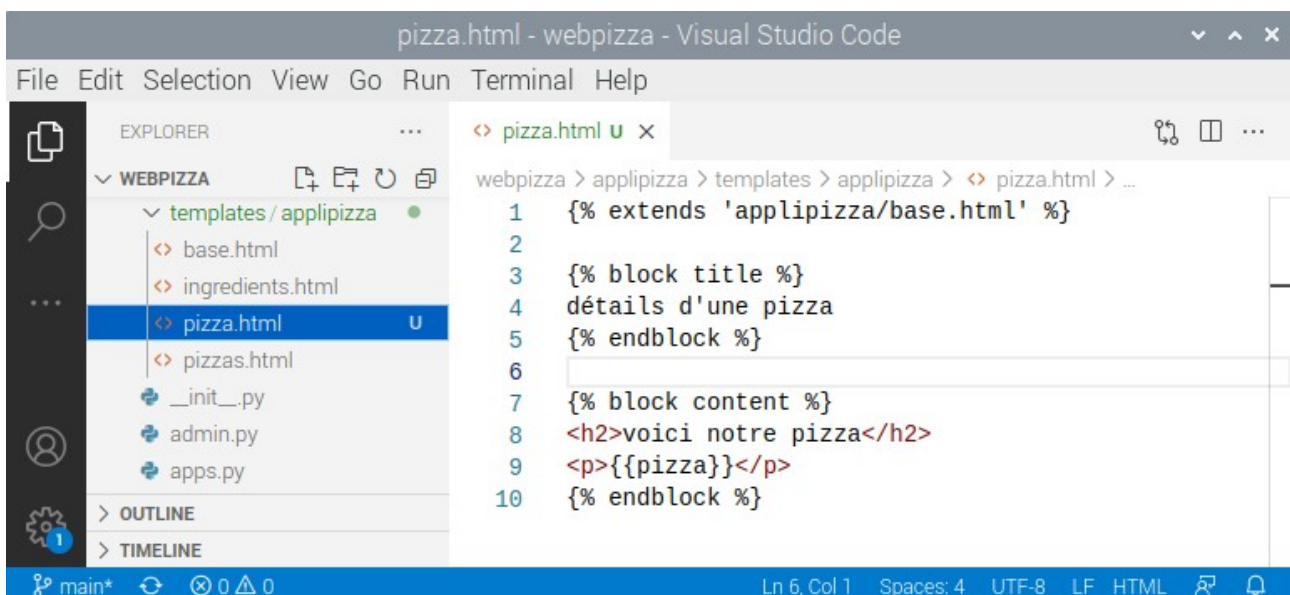
webpizza > applipizza > views.py > pizza
37
38 def pizza(request, pizza_id) :
39
40     # récupération de la pizza dont l'idPizza a été passé en
41     # paramètre dans l'url définie dans urls.py
42     laPizza = Pizza.objects.get(idPizza = pizza_id)
43
44     # on retourne l'emplacement de la vue et la pizza récupérée
45     return render(
46         request,
47         "applipizza/pizza.html",
48         {"pizza" : laPizza}
49     )
50
51
```

Vous noterez la requête pour récupérer LA pizza identifiée par son `idPizza`

```
laPizza = Pizza.objects.get(idPizza = pizza_id)
```

et comme précédemment la façon de transmettre au template l'information concernant cette pizza.

3. Créez maintenant le template `pizza.html` avec le code suivant. Analysez ce code, vous y retrouverez des éléments déjà vus.



```
pizza.html - webpizza - Visual Studio Code
File Edit Selection View Go Run Terminal Help

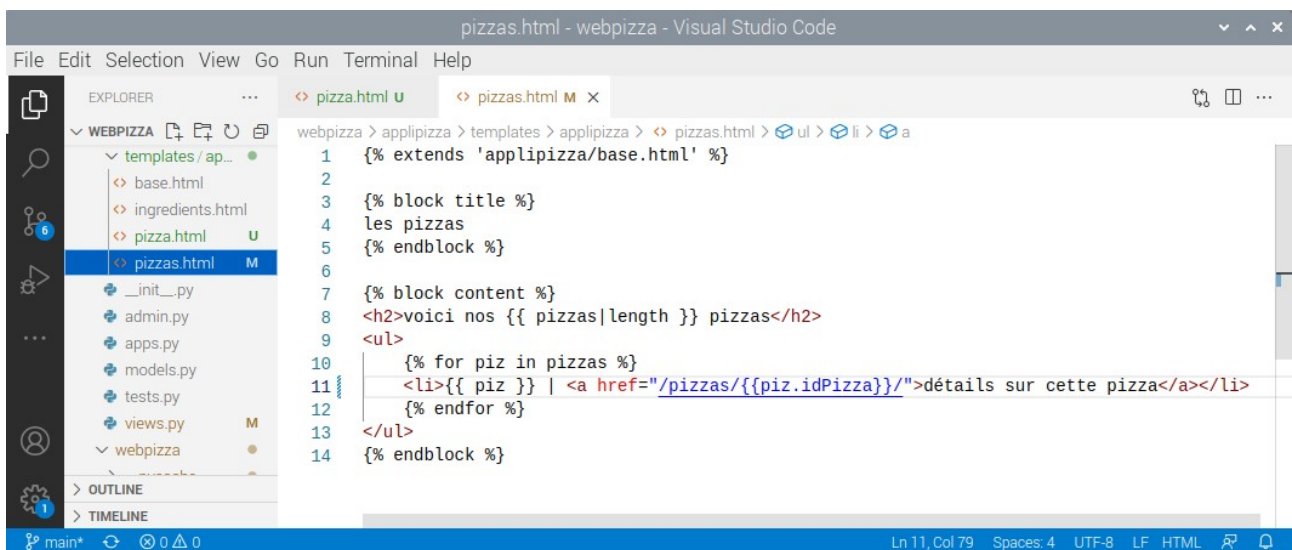
EXPLORER
  WEBPIZZA
    templates/applipizza
      base.html
      ingredients.html
      pizza.html U
      pizzas.html
    __init__.py
    admin.py
    apps.py
  OUTLINE
  TIMELINE

webpizza > applipizza > templates > applipizza > pizza.html > ...
1 {% extends 'applipizza/base.html' %}
2
3 {% block title %}
4 détails d'une pizza
5 {% endblock %}
6
7 {% block content %}
8 <h2>voici notre pizza</h2>
9 <p>{{pizza}}</p>
10 {% endblock %}
```

4. Essayez maintenant les deux url du 1. Vous devriez obtenir un résultat de ce type :



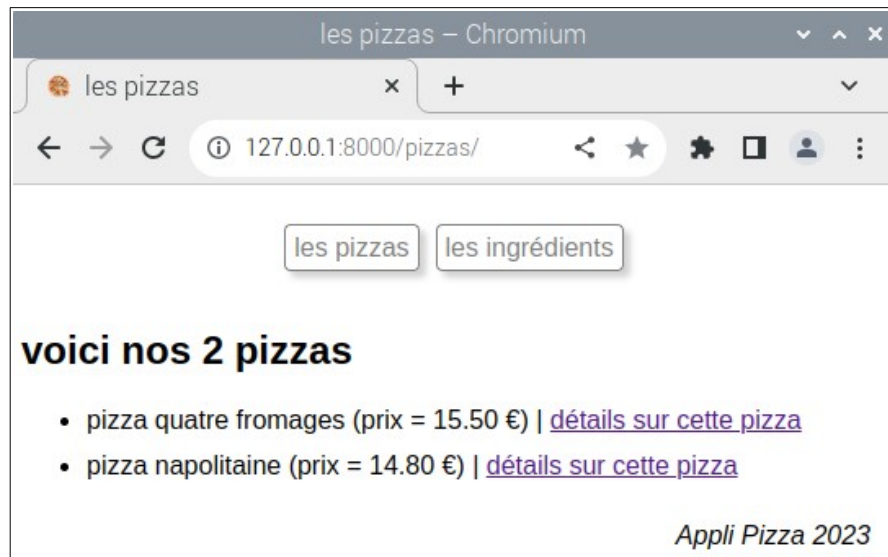
5. Nous allons maintenant ajouter, dans la page pizzas.html, les liens de détail menant à la page relative à chaque pizza. Changez le code pour intégrer ces liens :



6. Vérifiez l'effet. Pour le moment, cela ne présente pas vraiment d'intérêt, puisque la vue de détail n'apporte rien de plus que les renseignements déjà donnés par la page pizzas.html.

Mais cela va bientôt changer, puisque vous allez compléter cette vue de détail d'une pizza en listant les ingrédients entrant dans sa composition.

Rendu actuel de la page pizzas.html :



## Amélioration de la vue détail d'une pizza

Cette partie est à faire en autonomie.

Vous devrez améliorer la « vue » d'une pizza, définie dans `views.py`, pour qu'elle donne aussi la liste des ingrédients de la pizza, avec leur quantité.

Cette amélioration demandera de faire une requête sur la base de données, en récupérant de la table `Composition` toutes les entrées pour lesquelles l'identifiant de la pizza correspond à celui passé en argument (`pizza_id`).

Cette requête utilise la méthode « filter » :

```
compo = Composition.objects.filter(pizza = pizza_id)
```

- `pizza` désigne la clé étrangère de la classe `Composition`,
- `pizza_id` est le paramètre passé dans l'url.

Puis vous devrez construire une liste des ingrédients, que vous transmettez au template par l'intermédiaire du dictionnaire python dans lequel il n'y a pour le moment que la pizza à afficher.

Enfin, au niveau du template, vous devrez présenter les ingrédients proprement, par exemple sous forme d'un tableau.

Voici un exemple de rendu :

détails d'une pizza – Chromium

détails d'une pizza x +

127.0.0.1:8000/pizzas/1/

les pizzas les ingrédients

## voici notre pizza

- pizza quatre fromages (prix = 15.50 €)

## les 7 ingrédients de la pizza quatre fromages

| ingrédient        | quantité             |
|-------------------|----------------------|
| gorgonzola        | 75 grammes           |
| fromage de chèvre | 75 grammes           |
| parmesan          | 30 grammes           |
| mozzarella        | 1 boule              |
| crème fraîche     | 1 cuillerée à soupe  |
| oignon            | 1/2 oignon haché     |
| coulis de tomate  | 4 cuillerées à soupe |

Appli Pizza 2023