

## TP6 – Premiers formulaires

### Introduction, Contexte et Objectifs

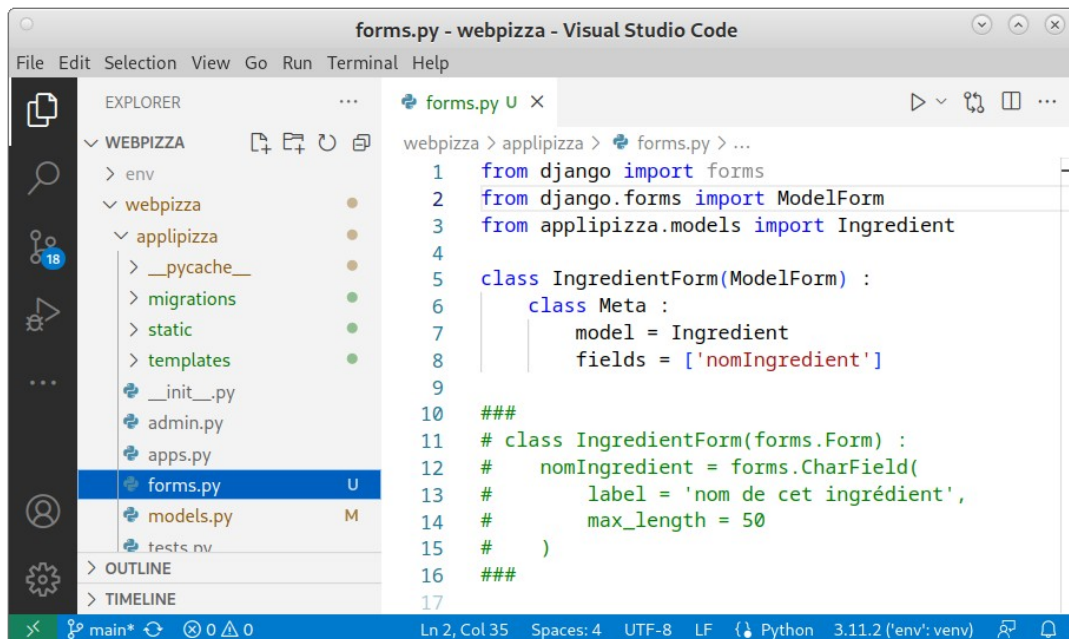
Pour le moment nous accédons à la base de données, et nous consultons les données, pour affichage. C'est de la « lecture seule ». L'idée est ici de :

- construire un formulaire pour récupérer les données de futurs ingrédients,
- traiter l'envoi du formulaire et ainsi avoir une action d'écriture sur la base de données. Cette action se fera sans être connecté au site. Autrement dit, tout le monde est « administrateur ». Nous verrons la connexion plus tard.

Il y a une méthode complètement automatique pour créer un formulaire adapté aux attributs d'une classe. Nous allons l'utiliser pour les formulaires de création d'une pizza et d'un ingrédient.

### Création du fichier forms.py

1. Au même niveau que views.py, créez un fichier forms.py. Nous allons y définir notre formulaire.
2. Dans ce fichier, importez la bibliothèque forms de django, puis de forms importez ModelForm, qui permet les formulaires automatiques. Créez la classe IngredientForm, qui hérite de ModelForm, et qui représentera le formulaire correspondant à un nouvel ingrédient. Le code, donné juste après, est vraiment des plus simples. La classe Meta fait le travail. En commentaires, la version non-automatique, où on déclare les champs équivalents à ceux de la classe.

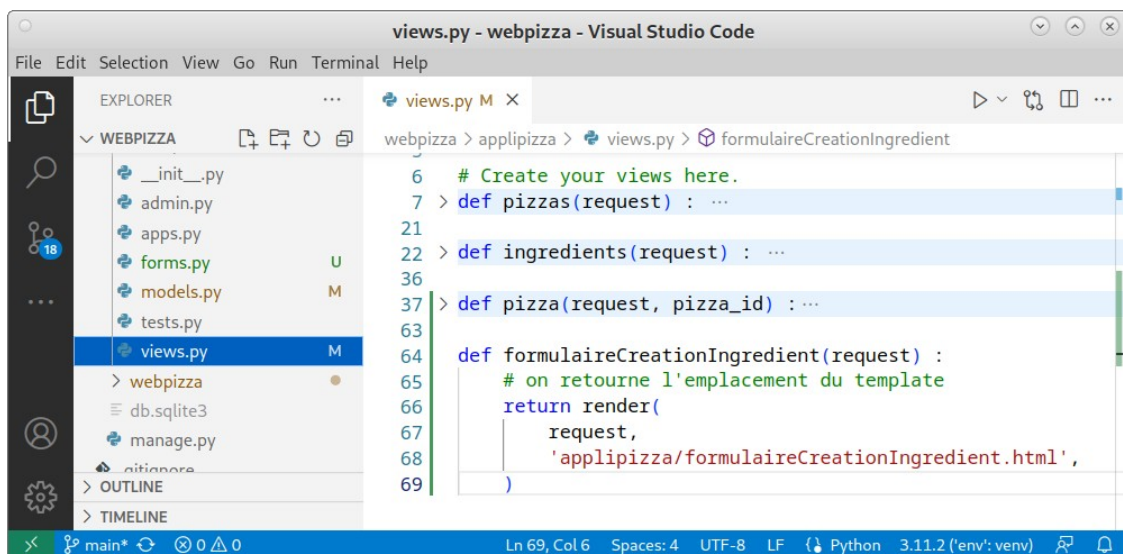


The screenshot shows the Visual Studio Code editor with the file `forms.py` open. The Explorer sidebar on the left shows the project structure: `WEBPIZZA` (containing `env`), `webpizza` (containing `__pycache__`, `migrations`, `static`, `templates`), and `applipizza` (containing `__init__.py`, `admin.py`, `apps.py`, `forms.py`, `models.py`, and `tests.py`). The `forms.py` file is selected. The main editor area shows the following Python code:

```
1 from django import forms
2 from django.forms import.ModelForm
3 from applipizza.models import Ingredient
4
5 class IngredientForm(ModelForm) :
6     class Meta :
7         model = Ingredient
8         fields = ['nomIngredient']
9
10    ###
11    # class IngredientForm(forms.Form) :
12    #     nomIngredient = forms.CharField(
13    #         label = 'nom de cet ingrédient',
14    #         max_length = 50
15    #     )
16    ###
17
```

## Création d'une view dans le fichier views.py

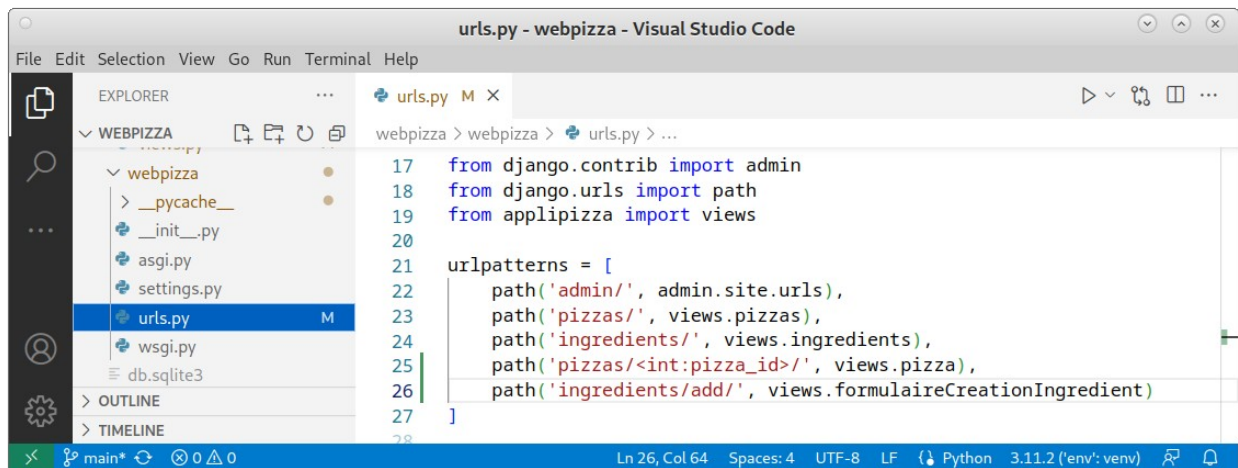
1. Créez dans `views.py` la view `formulaireCreationIngredient` qui retourne un render qui indique un template (pas encore créé) **`applipizza/formulaireCreationIngredient.html`** :



The screenshot shows the Visual Studio Code editor with the file `views.py` open. The Explorer sidebar on the left shows the project structure, with `views.py` selected under `applipizza`. The main editor area shows the following Python code:

```
6 # Create your views here.
7 > def pizzas(request) : ...
21
22 > def ingredients(request) : ...
36
37 > def pizza(request, pizza_id) : ...
63
64 def formulaireCreationIngredient(request) :
65     # on retourne l'emplacement du template
66     return render(
67         request,
68         'applipizza/formulaireCreationIngredient.html',
69     )
```

2. Dans le fichier `urls.py`, définissez un path qui permet d'accéder à ce formulaire. Ce path pourra être de la forme `ingredients/add` et appellera la « vue » précédente.

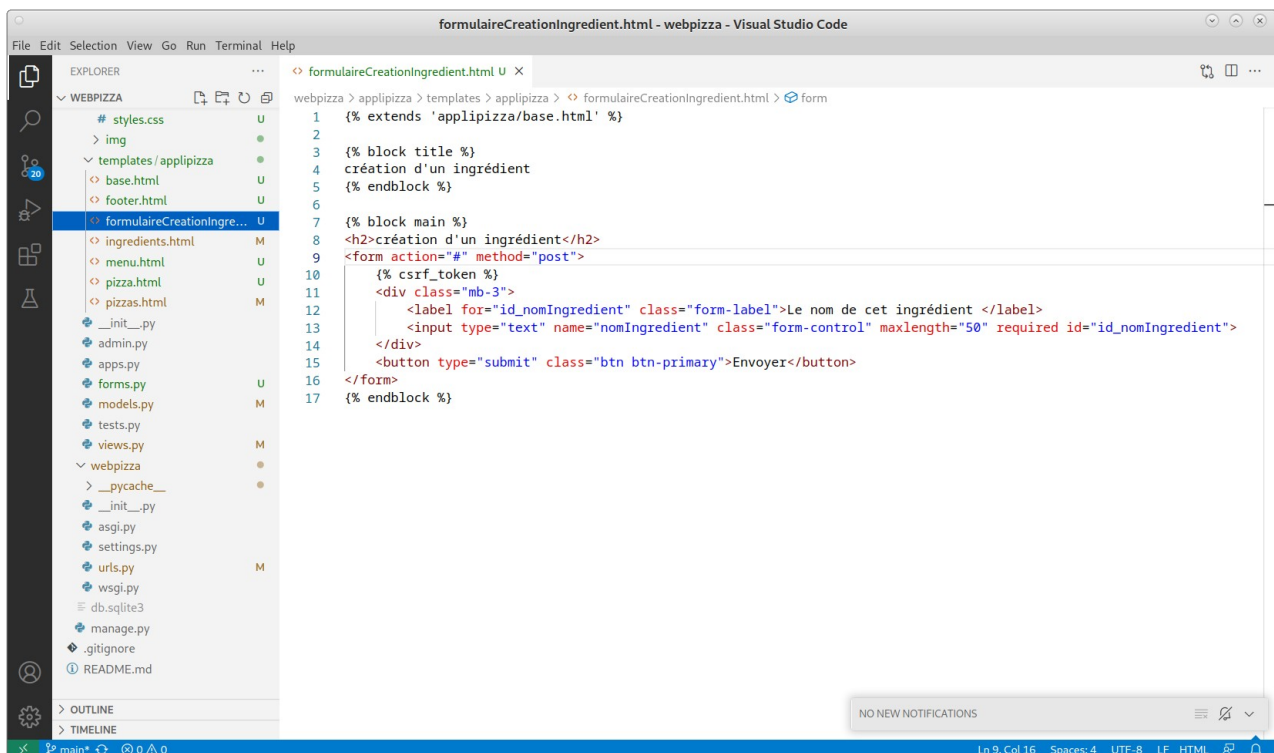


```

urls.py - webpizza - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
webpizza
  __pycache__
  __init__.py
  asgi.py
  settings.py
  urls.py
  wsgi.py
  db.sqlite3
  OUTLINE
  TIMELINE
webpizza > webpizza > urls.py > ...
17 from django.contrib import admin
18 from django.urls import path
19 from applipizza import views
20
21 urlpatterns = [
22     path('admin/', admin.site.urls),
23     path('pizzas/', views.pizzas),
24     path('ingredients/', views.ingredients),
25     path('pizzas/<int:pizza_id>', views.pizza),
26     path('ingredients/add/', views.formulaireCreationIngredient)
27 ]
Ln 26, Col 64 Spaces: 4 UTF-8 LF Python 3.11.2 (env: venv)

```

3. Dans le répertoire `template/applipizza`, créez un fichier `formulaireCreationIngredient.html` qui aura la même base que les autres :



```

formulaireCreationIngredient.html - webpizza - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
webpizza
  styles.css
  img
  templates/applipizza
    base.html
    footer.html
    formulaireCreationIngred...
    ingredients.html
    menu.html
    pizza.html
    pizzas.html
  __init__.py
  admin.py
  apps.py
  forms.py
  models.py
  tests.py
  views.py
  webpizza
    __pycache__
    __init__.py
    asgi.py
    settings.py
    urls.py
    wsgi.py
    db.sqlite3
    manage.py
    .gitignore
    README.md
  OUTLINE
  TIMELINE
webpizza > applipizza > templates > applipizza > formulaireCreationIngredient.html > form
1 {% extends 'applipizza/base.html' %}
2
3 {% block title %}
4 création d'un ingrédient
5 {% endblock %}
6
7 {% block main %}
8 <h2>création d'un ingrédient</h2>
9 <form action="#" method="post">
10     {% csrf_token %}
11     <div class="mb-3">
12         <label for="id_nomIngredient" class="form-label">Le nom de cet ingrédient </label>
13         <input type="text" name="nomIngredient" class="form-control" maxlength="50" required id="id_nomIngredient">
14     </div>
15     <button type="submit" class="btn btn-primary">Envoyer</button>
16 </form>
17 {% endblock %}
Ln 9, Col 16 Spaces: 4 UTF-8 LF HTML

```

On y retrouve le code html d'un formulaire avec des classes bootstrap.

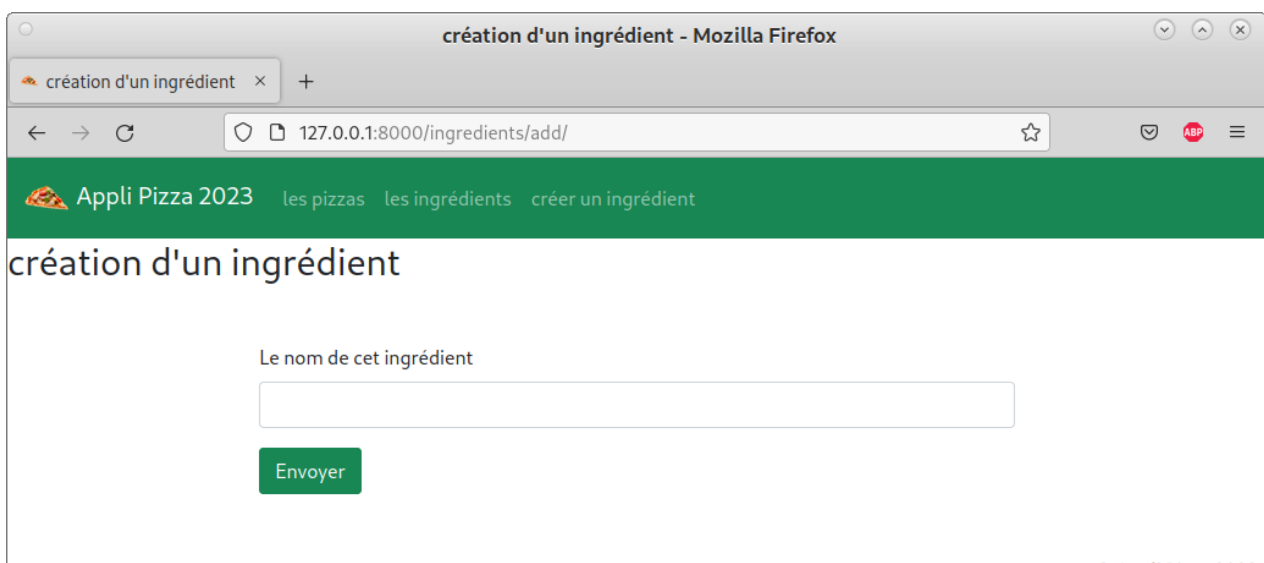
Le « csrf\_token » est un « jeton » qui sert à lutter contre les failles CSRF. Nous pouvons laisser cela de côté, ce n'est pas notre thème.

<https://www.leblogduhacker.fr/faille-csrf-explications-contre-mesures/>

Pour le moment, on ne renseigne pas l'action du formulaire. Plus tard, l'action sera la création du nouvel ingrédient.

Si tout est bien fait, l'url 127.0.0.1:8000/ingredients/add/ doit amener au formulaire, qu'on peut remplir, mais dont l'envoi ne donnera rien, puisque le traitement du formulaire n'a pas encore été programmé.

Pensez à ajouter un lien dans le menu pour amener au formulaire :



The screenshot shows a Mozilla Firefox browser window titled "création d'un ingrédient - Mozilla Firefox". The address bar shows the URL "127.0.0.1:8000/ingredients/add/". The page has a green header with a pizza icon and the text "Appli Pizza 2023", followed by navigation links: "les pizzas", "les ingrédients", and "créer un ingrédient". The main content area is titled "création d'un ingrédient" and contains a form with a single text input field labeled "Le nom de cet ingrédient" and a green "Envoyer" button below it.

## L'action lancée suite à l'envoi du formulaire

1. Créez dans views.py la view creerIngredient qui va gérer le traitement des informations reçues via le formulaire :

a. on récupère le formulaire :

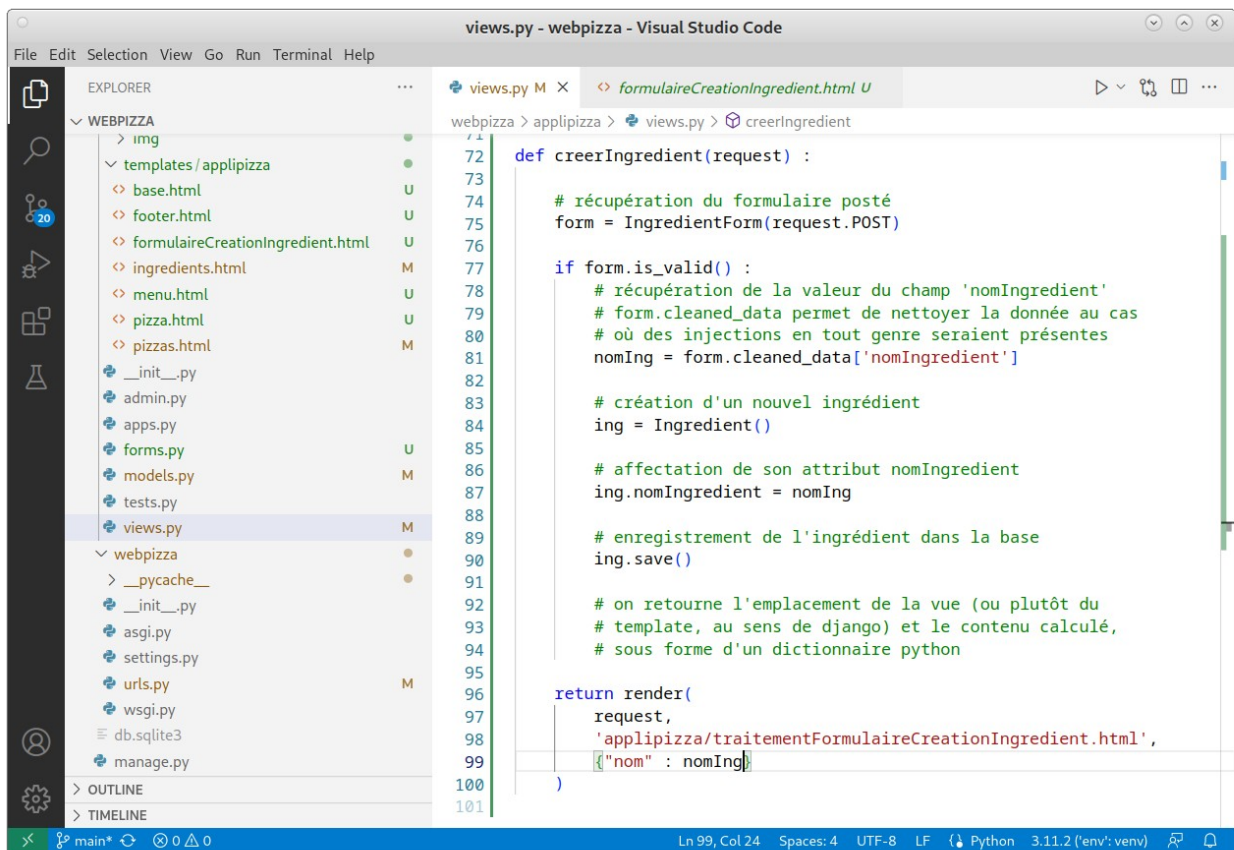
```
form = IngredientForm(request.POST)
```

- b. si le formulaire est « valide » (vérification automatique) alors on récupère le nom de l'ingrédient par :

```
nomIng = form.cleaned_data['nomIngredient']
```

- c. puis on crée une instance de la classe Ingredient, on lui donne comme nomIngredient la valeur récupérée, et on sauve cet Ingredient dans la base.

- d. Enfin, on retourne un render vers le template nommé applipizza/traitementFormulaireCreationIngredient.html, avec comme contenu le nom récupéré (template pas encore créé).



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

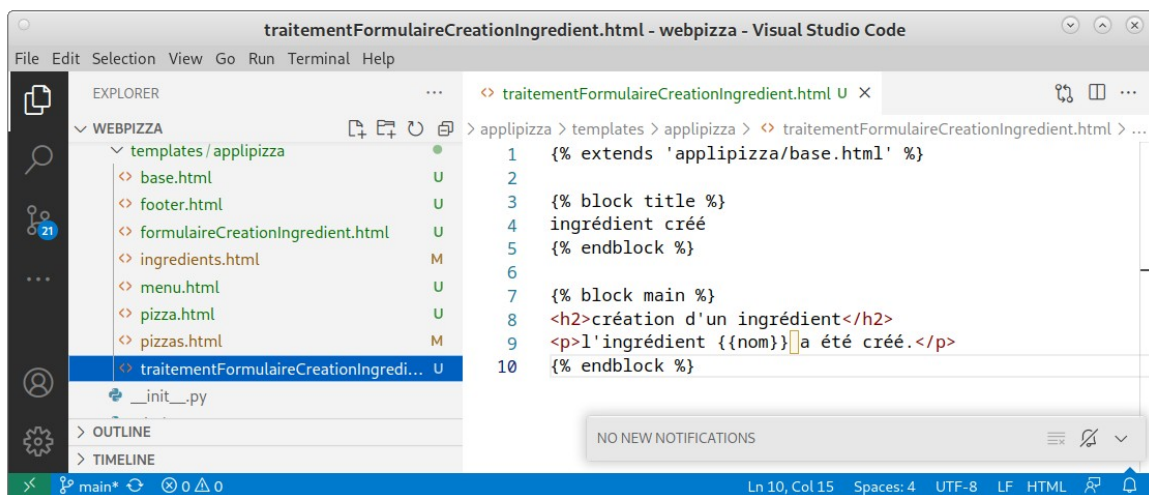
EXPLORER
webpizza
├── img
├── templates/applipizza
│   ├── base.html
│   ├── footer.html
│   ├── formulaireCreationIngredient.html
│   ├── ingredients.html
│   ├── menu.html
│   ├── pizza.html
│   └── pizzas.html
├── __init__.py
├── admin.py
├── apps.py
├── forms.py
├── models.py
├── tests.py
├── views.py
├── webpizza
│   ├── __pycache__
│   ├── __init__.py
│   ├── asgi.py
│   ├── settings.py
│   ├── urls.py
│   ├── wsgi.py
│   ├── db.sqlite3
│   └── manage.py
├── OUTLINE
└── TIMELINE

views.py
71
72 def creerIngredient(request):
73
74     # récupération du formulaire posté
75     form = IngredientForm(request.POST)
76
77     if form.is_valid():
78         # récupération de la valeur du champ 'nomIngredient'
79         # form.cleaned_data permet de nettoyer la donnée au cas
80         # où des injections en tout genre seraient présentes
81         nomIng = form.cleaned_data['nomIngredient']
82
83         # création d'un nouvel ingrédient
84         ing = Ingredient()
85
86         # affectation de son attribut nomIngredient
87         ing.nomIngredient = nomIng
88
89         # enregistrement de l'ingrédient dans la base
90         ing.save()
91
92         # on retourne l'emplacement de la vue (ou plutôt du
93         # template, au sens de django) et le contenu calculé,
94         # sous forme d'un dictionnaire python
95
96     return render(
97         request,
98         'applipizza/traitementFormulaireCreationIngredient.html',
99         {"nom": nomIng})
100
101
```

2. Dans le fichier urls.py, définissez un path qui permet d'accéder à la page annonçant que le traitement a bien été effectué. Ce path sera de la forme ingredients/create/ et appellera la view précédente.

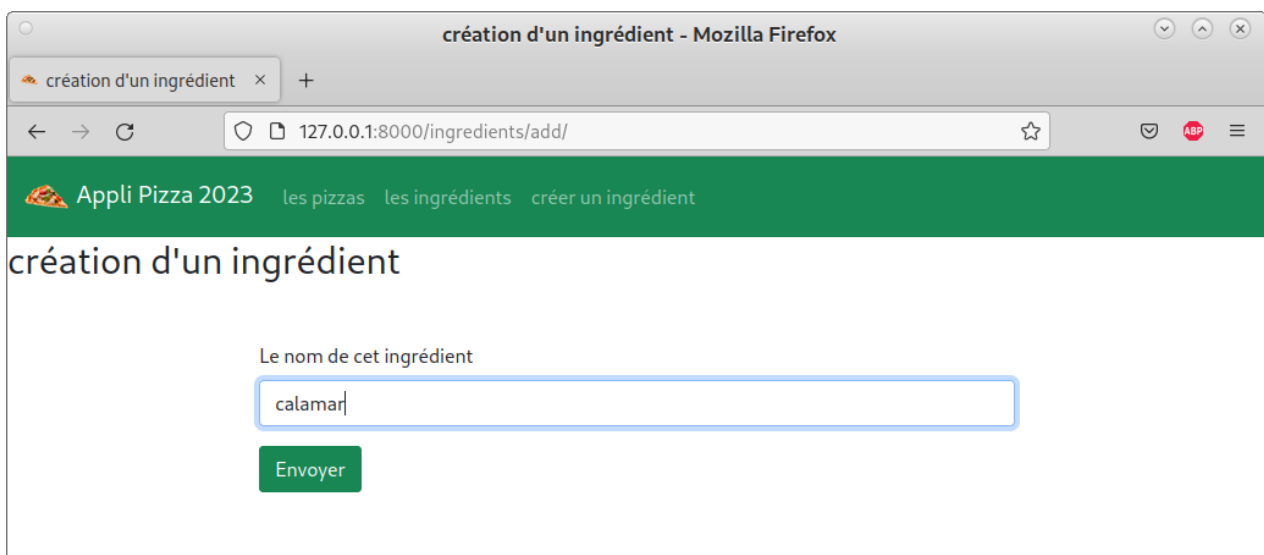
3. Dans le template formulaireCreationIngredient.html, indiquez maintenant la bonne action : /ingredients/create/

4. Créez `traitementFormulaireCreationIngredient.html`, template qui hérite de `base.html`, comme tous les autres, et prévient juste que la création a été effectuée, et utilisant la variable de gabarit `{{nom}}` qui permet de rappeler le nom du nouvel ingrédient :



```
traitementFormulaireCreationIngredient.html - webpizza - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
WEBPIZZA
  templates/applipizza
    < base.html U
    < footer.html U
    < formulaireCreationIngredient.html U
    < ingredients.html M
    < menu.html U
    < pizza.html U
    < pizzas.html M
    < traitementFormulaireCreationIngredi... U
  __init__.py
OUTLINE
TIMELINE
> applipizza > templates > applipizza > < traitementFormulaireCreationIngredient.html > ...
1 {% extends 'applipizza/base.html' %}
2
3 {% block title %}
4 ingrédient créé
5 {% endblock %}
6
7 {% block main %}
8 <h2>création d'un ingrédient</h2>
9 <p>l'ingrédient {{nom}} a été créé.</p>
10 {% endblock %}
```

5. Testez que tout fonctionne. Vous pourrez aussi contrôler, dans `sqlitebrowser`, l'enrichissement progressif de la table des ingrédients.



création d'un ingrédient - Mozilla Firefox

création d'un ingrédient x +

127.0.0.1:8000/ingredients/add/

Appli Pizza 2023 les pizzas les ingrédients créer un ingrédient

### création d'un ingrédient

Le nom de cet ingrédient

Envoyer

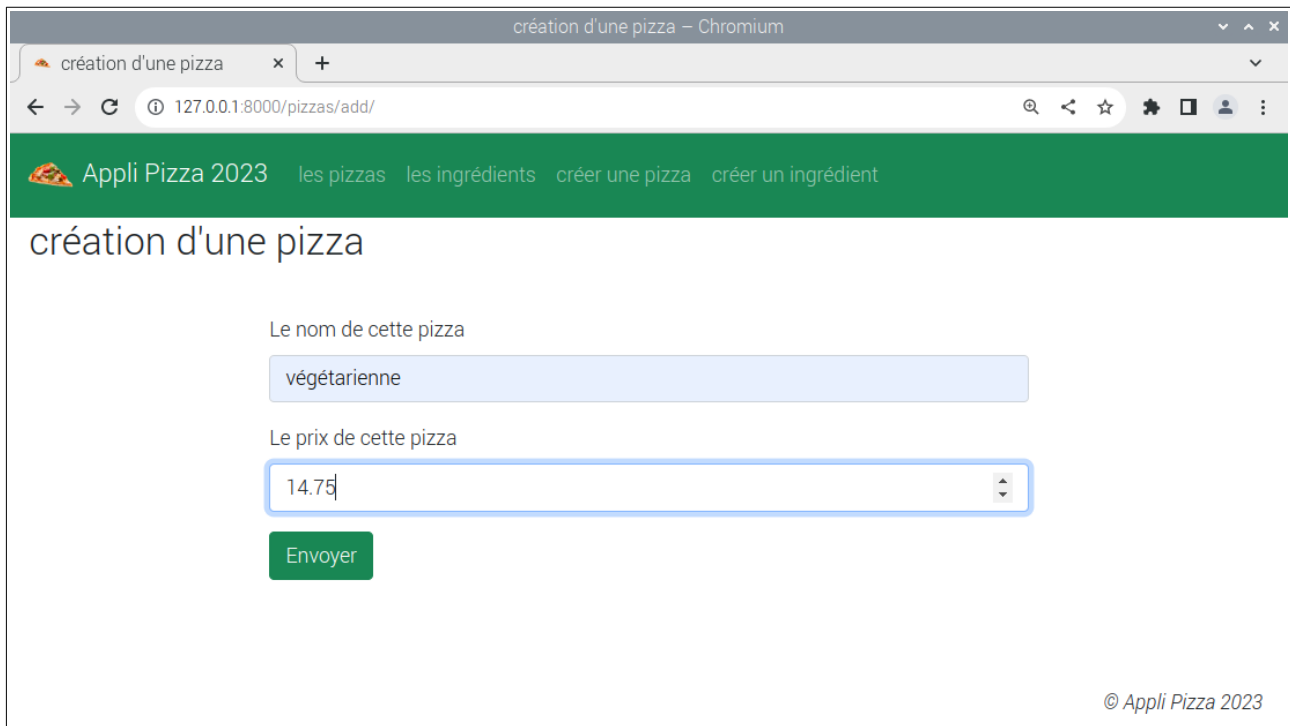


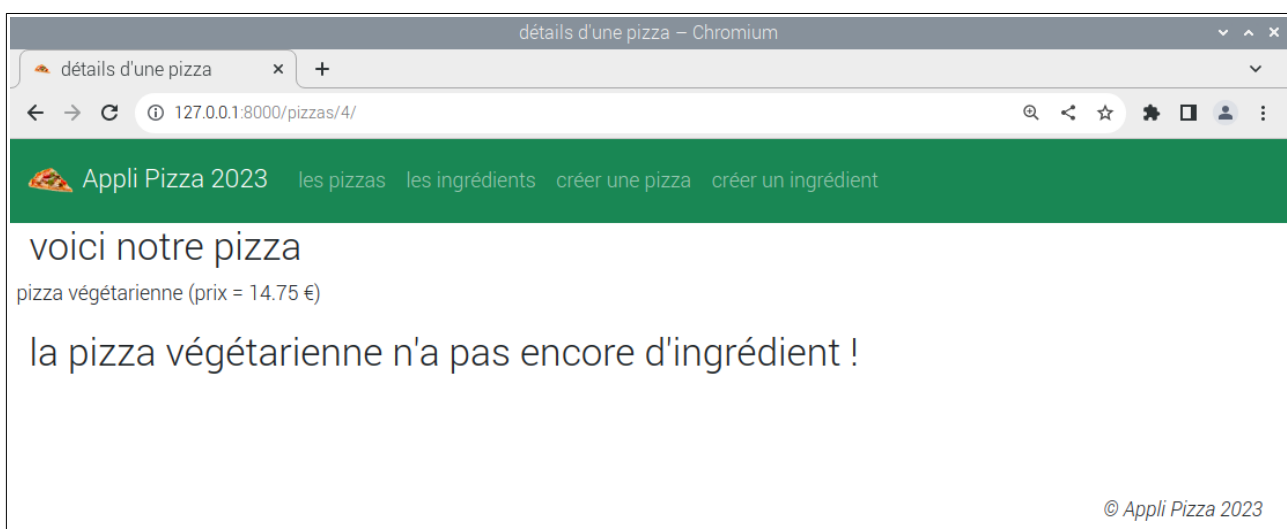
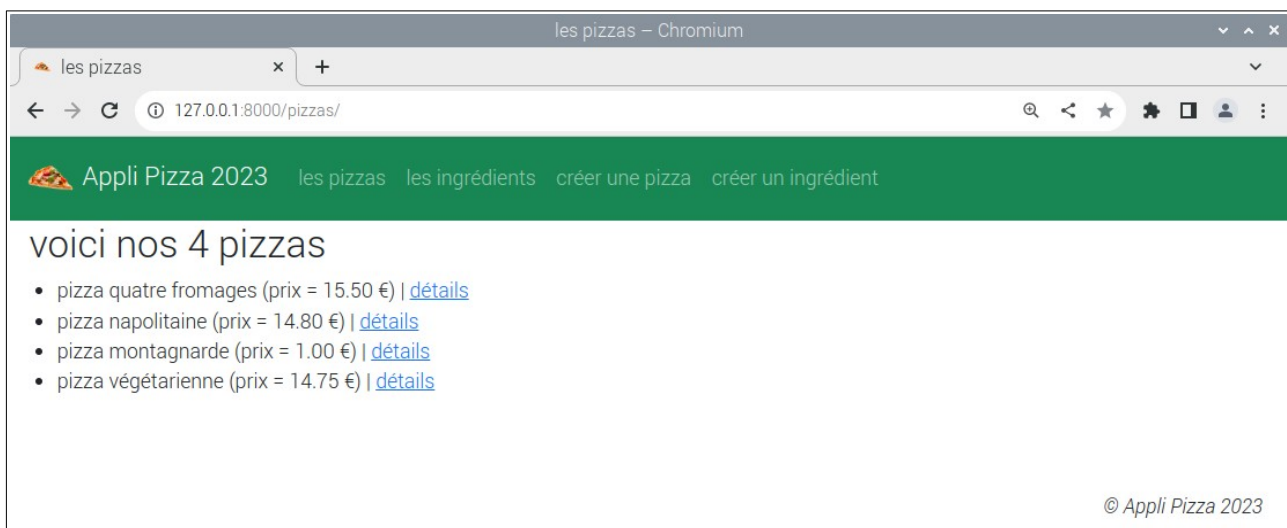
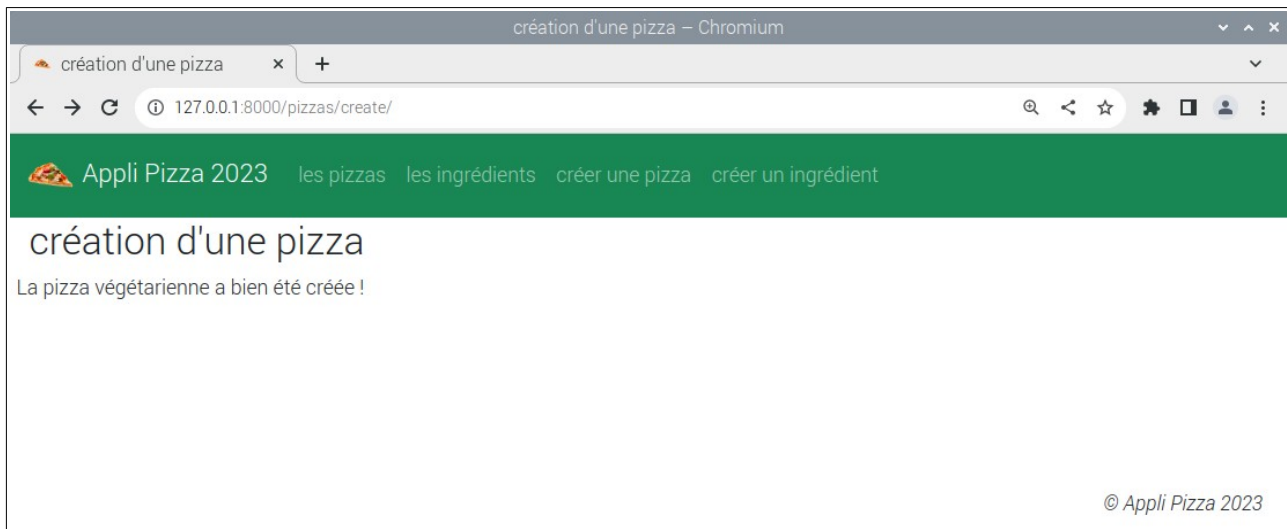


## Avez-vous bien tout assimilé ?

Recommencez tout, cette fois pour la création d'une nouvelle pizza (nom, et prix).

Voici un exemple de rendu :





Remarque : pour la dernière capture, il y a eu un choix : s'il n'y a pas encore d'ingrédient, alors ...