4.1 Les formulaires avec Flask

a. Introduction

Les formaulaires sont une partie essentielle à toute web application. Ils constitutent un moyen efficace d'échanger et d'intéragir avec l'utilisateur.
Que ce soit pour se connecter ou soumettre des données dans bien des cas, il est nécessaire d'incorporer des formulaires. Il est indispensable de valider ces données pour des raisons de cohérence mais aussi de sécurité.
Le module `WTForms` est celui qui a en charge la gestion des formulaires. Il fourni beaucoup de validations de champs, côté serveur, par défaut et facilite donc la réalisation de formulaire.
Installation de `WTForms`
`pip install Flask-WTF`

b. Quickstart

Pour ce premier exemple, on reprend la situation présentée dans la documentation `https://flask-wtf.readthedocs.io/`

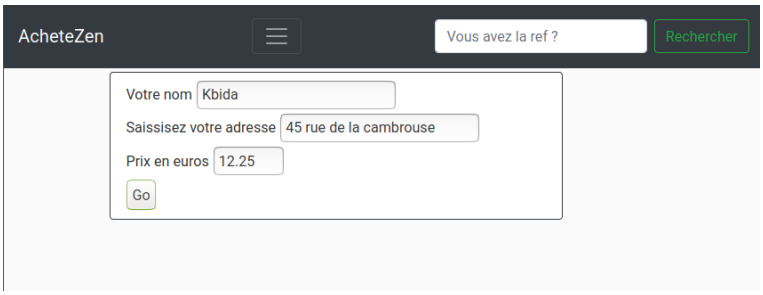
1. On crée un premier fichier `forms.py` qui contiendra le modèle de nos formulaires

```
from flask_wtf import FlaskForm
from wtforms import StringField, DecimalField
from wtforms.validators import DataRequired, Length


class MyForm(FlaskForm):
    name = StringField('Votre nom', validators=[DataRequired()])
    address = StringField('Saissisez votre adresse',
                          validators=[DataRequired(), Length(max=200)])
    price = DecimalField('Prix en euros', validators=[DataRequired()])
```

2. On crée une vue dans le dossier `templates` contenant le formulaire HTML `monformulaire.html`

```
<form method="POST" action="">
    {{ form.hidden_tag() }}
    {{ form.name.label }} {{ form.name(size=20) }}
    {{ form.address.label }} {{ form.adress(size=75) }}
    {{ form.price.label }} {{ form.price(size=10) }}
    <input type="submit" value="Go">
</form>
```

`{{ form.hidden_tag() }}` permet de générer un jeton de sécurité pour certifier l'origine du formulaire.
Cela devrait donner cela :



3. On va récupérer les données du formaulaire dans le controller `app.py` pour cela : on importe le modèle du formulaire `Myform`

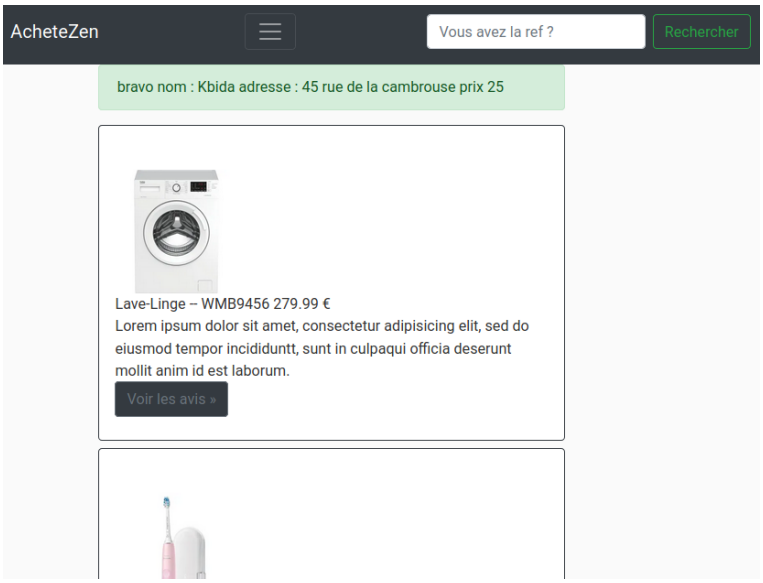
```
from flask import Flask, render_template, flash, redirect
from forms import MyForm

@app.route('/submit', methods=['GET', 'POST'])
def submit():
    form = MyForm()
    if form.validate_on_submit():
```

```
        flash('bravo nom : {0} adresse : {1} prix {2}'.format(form.name.data,
                                                              form.address.data,
                                                              form.price.data))
        return redirect('/home')
    return render_template('submit.html', title='Test formulaire', form=form)
```

Pour visualiser le résultat dans `home` on peut rajouter un affichage de message `flash` dans la vue `layout.html` avant le `block content`

```
{% with messages = get_flashed_messages(with_categories=true) %}
    {% if messages %}
        {% for category, message in messages %}
            <div class="alert alert-success">
              {{ message }}
            </div>
        {% endfor %}
    {% endif %}
{% endwith %}
```

Cela devrait donner cela :



c. À vous de créer votre propre formulaire

Ce formulaire est constitué :

- d'un sélecteur pour les utilisateurs

```
<select ... >
   <option value= ...> </option>
   <option value= ...> </option>
</select>
```

- d'un sélecteur pour les produits
- d'un champ de texte pour le contenu du post

```
<textarea ...> </textarea>
```

- un curseur de type numérique permettant d'attribuer une note de 0 à 5

```
<input type="range" id="rating" name="rating"
        min="0" max="5">
<label for="rating"
>Note de 0 à 5</label>
```

- d'un bouton pour valider l'avis

```
<button type="submit">Valider</button>.
```