Micro site web en python

Réalisé à l'aide du package Flask (https://pypi.org/project/Flask/).

Attention: si vous executez la cellule ci dessous, elle ne s'arrêtera pas. Pour l'interrompre dans jupyterlab: Kernel > Interrupt Kernel.

Quand le site s'exécute, rendez vous sur http://127.0.0.1:5000 (http://127.0.0.1:5000) pour le consulter.

(127.0.0.1 est l'adresse IP de votre propre ordinateur)

Entrée [ ]:

```
# This small program runs a website with two pages:
# - GET / => returns list of items + display a form to add new item
# - POST /add-item => add a new item, display a link to main page

from flask import Flask, request
import json

site = Flask('site')

todo_list = []


@site.route('/')
def home():
    # On regarde si la page web a été demandée avec le paramètre &format=json
    # dans l'url. Si c'est le cas: on renvoie la liste au format JSON.
    format = request.args.get('format', 'html')
    if format == 'json':
        return json.dumps(todo_list)

    # Sinon: on génère une page HTML contenant la liste:
    items = [f'<li>{item}</li>' for item in todo_list]
    return f"""
        <h1>Todo List</h1>
        <ul>
            {''.join(items)}
        </ul>
        <form action="/add-item" method="post">
            <input type="text" name="newitem" />
            <input type="submit" value="ajouter" />
        </form>
    """


@site.route('/add-item', methods=['POST'])
def add_item():
    item = request.form['newitem']  # retrieve the data sent by the browser
    todo_list.append(item)

    return f"""
        <p>
            {item} has been added to todo-list !<br><br>
            Go <a href="/">back</a> to main page.
        </p>
    """


site.run()
```

Interagir avec le site web flask-website

Préalable: il faut que le site s'exécute (cf. le notebook jupyterlab run-flask-website.ipynb)

Entrée [1]:

```
# Récupération de la page d'accueil:
requests.get('http://127.0.0.1:5000').text
```

Out[1]:

```
'\n        <h1>Todo List</h1>\n        <ul>\n            \n        </u
l>\n        <form action="/add-item" method="post">\n            <inpu
t type="text" name="newitem" />\n            <input type="submit" valu
e="ajouter" />\n        </form>\n    '
```

Entrée [2]:

```
# pas la bonne méthode HTTP (GET au lieu de POST)
requests.get('http://127.0.0.1:5000/add-item')
```

Out[2]:

```
<Response [405]>
```

Entrée [3]:

```
# Bonne méthode, mais on a oublié d'envoyer des data:
requests.post('http://127.0.0.1:5000/add-item')
```

Out[3]:

```
<Response [400]>
```

Entrée [4]:

```
resp = requests.post('http://127.0.0.1:5000/add-item', data={'newitem': 'biere'})
```

Entrée [5]:

```
resp
```

Out[5]:

```
<Response [200]>
```

Entrée [6]:

```
resp.text
```

Out[6]:

```
'\n        <p>\n            biere has been added to todo-list !<br><br
>\n            Go <a href="/">back</a> to main page.\n        </p>\n
'
```

Entrée [7]:

```
requests.post('http://127.0.0.1:5000/add-item', data={'newitem': 'pizza'})
```

Out[7]:

```
<Response [200]>
```

Entrée [8]:

```
from bs4 import BeautifulSoup

html = requests.get('http://127.0.0.1:5000').text
soup = BeautifulSoup(html)
```

Entrée [9]:

```
soup.find_all('li')
```

Out[9]:

```
[<li>biere</li>, <li>pizza</li>]
```

Entrée [10]:

```
[elem.text for elem in soup.find_all('li')]
```

Out[10]:

```
['biere', 'pizza']
```

Pourquoi s'embêter à récupérer du HTML alors que je peux demander à la page web de me renvoyer du JSON ?

Entrée [11]:

```
response = requests.get('http://127.0.0.1:5000?format=json')
```

Entrée [12]:

```
import json
liste = json.loads(response.text)
```

Entrée [13]:

```
type(liste)
```

Out[13]:

```
list
```

Entrée [14]:

```
liste
```

Out[14]:

```
['biere', 'pizza']
```