TP1 - Mise en place d'un site web en local

L'objectif du TP est de mettre en place votre premier site web en local. Votre site sera accessible depuis une application cliente, un navigateur web.

Pour cela, nous allons utiliser le module Flask disponible sous Python. Ce module permet de créer un serveur web en local.

Création du formulaire d'ajout d'un fil de discussion (sujet)

Vous allez créer un formulaire web qui permettra à l'utilisateur de créer un nouveau fil de discussion dans le forum.

Un formulaire web est une interface homme machine (IHM). L'utilisateur devra compléter le formulaire et valider en cliquant sur un bouton qui transmettra les données au serveur Web.

Rappel :

Pour créer un formulaire, il faut utiliser la balise <form> pour rappel.

Pour créer des entrées pour l'utilisateur (texte à saisir, choix dans une liste,...), vous pouvez utiliser la balise <input>. Je vous invite à vous référer au polycopié de la séquence IHM sur le Web.

Exemple de formulaire Web :

```
<!DOCTYPE html>
<html lang="fr">

    <head>
        <title>Forum NSI - Sujet</title>
        <meta charset="UTF-8" />
        <link type="text/css" rel="stylesheet" href="/static/style.css"/>
    </head>

    <body>
        <div id="home">
            <div class="banner">
                    <h1>Création d'un nouveau sujet</h1>
                    <a href="index.html">Accueil</a>
            </div>
            <div>
                <form action="/creerSujet" method="POST" >
                    <input type="label" value="Nom du sujet:" />
                    <input type="text" id="nom" name="nom"/>
                    <input type="file" accept="png" />
                    <input type="submit" value="Créer le sujet"/>

                </form>
            </div>

    </body>

</html>
```

Activité 1 - Formulaire Web de test

1. Créer un formulaire de test en reprenant le code HTML ci-dessus. Votre formulaire se nommera «sujet.html »
2. Afficher le formulaire dans un navigateur Web pour vérifier le rendu

Activité 2 – Créer une page web de traitement de la réponse

Nous allons maintenant créer une page web qui sera complétée par le serveur suite à la création du fil de discussion.

1. Créer un fichier qui se nommera « reponse.html »
2. Recopier dans le fichier créé le code HTML ci-dessous.

```
<!doctype html>
<html  lang="fr-FR">
  <head>
    <meta charset="utf-8">
  </head>
  <body>
        <h2>{{reponse}} </h2>
        <h2><a href = "/">Retourner à la page d'accueil</a></h2>
</body>
</html>
```

Remarque : {{reponse}} permet d'afficher le paramètre nommé « reponse » dans le formulaire. Ce paramètre sera géré par le serveur Web. Nous verrons cela dans les activités suivantes.

Activité 3 - Création d'un serveur web local avec Python

Nous allons utiliser le module Flask. La semaine dernière vous avez installé l'environnement de travail à savoir :

- L'IDE (Environnement de développement) Thonny en local
- Le module Flask

Se référer au manuel d'installation.

1. Lancer votre IDE Thonny
2. Créer un dossier «projetForum»
3. Créer un module Python que vous nommerez « projetWeb.py » et enregistrer le module dans le dossier créé précédemment.

4 Écrire dans le module créé le programme suivant :

```
from flask import Flask, render_template, request
import sorting_hat as sh

app = Flask(__name__)

@app.route('/')
def index():
   return render_template('index.html')

@app.route('/valide', methods=["POST"])
def valide():
   reponse = request.form['test']
   print(reponse)
   return render_template('reponse.html',reponse=reponse)
```

Explication du programme :

app = Flask(__name__) permet de créer un serveur Web en local, c'est-à-dire que le serveur sera sur votre machine. Il ne s'agit pas d'un serveur distant que vous pourriez partager avec l'ensemble de vos camardes. Vous aurez chacun votre serveur Web en local.

```
@app.route('/')
def index():
   return render_template('index.html')
```

Cette partie du code permet d'indiquer que lorsqu'on appellera la « route » / c'est-à-dire lorsque l'on va demander à accéder au site web qui sera créé en lançant le programme à savoir http://127.0.0.1:5000/, le serveur **recevra une requête HTTP de type « GET »** pour accéder à la page d'accueil du site. La fonction index() sera alors exécutée par le serveur.
return render_template('index.html') permet de renvoyer au client le contenu de la page index.html.

```
@app.route('/valide', methods=["POST"])
def valide():
   reponse = request.form['test']
   print(reponse)
   return render_template('reponse.html',reponse=reponse)
```

Cette partie du code sera exécutée lorsque le client, le navigateur web, transmettra une requête HTTP avec une action « /valide » et la méthode « POST ».
Concrètement lorsque l'utilisateur aura terminé de sélectionner ou saisir les données du formulaire de notre page « index.html », il cliquera sur le bouton « Envoyer ». Cela aura pour effet de transmettre la requête HTTP au serveur.
Vous verrez que dans notre formulaire Web, nous avons `<form method="post" action="/valide">`. Ici, nous avons indiqué que

nous transmettrons une requête http avec l'action et la méthode indiquée qui correspond à notre code du dessus.

reponse = request.form['test'] permet de récupérer la saisie de l'utilisateur dans le champ (<input>) avec le nom « test » (propriété name).

return render_template('reponse.html',reponse=reponse) permet au serveur de renvoyer le contenu de la page reponse.html avec le paramètre « reponse ». La valeur de ce paramètre pourra être affiché edans la page HTML grâce au code vu lors de la création du formulaire : {{reponse}}. En fait, notre serveur web va complété la page « reponse.html » en remplaçant {{reponse}} par la valeur du paramètre reponse.

5 Créer dans votre répertoire «projetForum» un répertoire « templates ».
6 Copier dans le répertoire templates vos deux pages HTML :
   - index.html
   - reponse.html
7 Exécuter votre programme

Vous devriez avoir le retour suivant dans la console :

```
>>> %Run projetWeb.py

   # Running the app with options chosen by Thonny. See Help for details.
   * Serving Flask app "projetWeb" (lazy loading)
   * Environment: development
   * Debug mode: off
   * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

8 Cliquer sur l'url http://127.0.0.1:5000

La page d'accueil, index.html, devrait s'afficher dans votre navigateur web.
Si vous saisissez du texte dans le champ puis vous cliquez sur « envoyer » vous devriez voir apparaitre la page HTML « reponse.html » avec votre saisie.

9 Observer les traces disponibles dans la console de votre IDE Thonny.
Vous devriez voir apparaître les requêtes HTTP reçues par votre serveur.

Remarque enseignant : Cela sera illustré en cours par le test du programme d'un des élèves. Ce document n'est pas exhaustif et mériterait d'être complété par une fiche indiquant comment insérer du « code Python » dans une page HTML notamment avec une boucle for.