TD9 : structuration d'un site web

UCBL - Département Informatique de Lyon 1 – BDW - automne 2024

Objectif du TD : structurer un site web avec l'architecture MVC

Rappel du schéma relationnel :

Activités	(idA, *titre*, *type*, #*idU*)
Semaines	(année, numéro, *ouverteAuVote*)
Utilisatrices	(idU, *nom*, *prénom*, *email*)
Voter	(#*idA*, date, #*idU*, #*année*, #*numéro*, *aAimé*)

fichier 1

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8" />
  <link type="text/css" rel="stylesheet"
  ↪  href="td_bdw/static/css/style.css">
  <title>{{ SESSION['APP'] }} - {{ SESSION['BASELINE']
  ↪  }}</title>
</head>

<body>
  {% include 'header.html' %}
  <div>
    {% include 'menu.html' %}
    <main>
      {% block main_content %}{% endblock %}
    </main>
  </div>
  {% include 'footer.html' %}
</body>
</html>
```

fichier 2

```
from datetime import datetime

SESSION['APP'] = "Activotons"
SESSION['BASELINE'] = "Votez pour vos activités
↪  préférées !"
SESSION['CURRENT_YEAR'] = datetime.now().year
```

fichier 3 (extrait)

```
def get_distinct_categories(connexion):
  with connexion.cursor() as cursor:
    try:
      @@@@.execute("SELECT DISTINCT(@@@@) FROM @@@@")
      return cursor.@@@@()
    except psycopg.Error as e:
      print(f"Error : {e}")
  return None
```

fichier 4 (extrait)

```
{% extends "base.html" %}

{% block main_content %}
<h1>Bienvenue sur {{ SESSION['APP'] }}</h1>
...
<section>
    <h2>Rechercher une activité</h2>
    <form method="post" action="/rechercher">
        <label for="champ_requete">Activité : </label>
        <input type="text" name="champ_requete"
        ↪  id="champ_requete" placeholder="cinéma"
        ↪  required><br>
        <label for="champ_categorie">Catégorie : </label>
        <select name="champ_categorie" id="champ_categorie">
            {% @@@@ cat in @@@@ %}
                <option>{{ @@@@ }}</option>
            {% endfor %}
        </select>
        <br/>
        <input type="submit" name="bouton_rechercher"
        ↪  value="Rechercher">
    </form>
</section>
```

fichier 5

```
from model.model_pg import get_distinct_categories
REQUEST_VARS['categories'] = @@@@(SESSION['CONNEXION'])
```

fichier 6 (extrait)

```
[[routes]]
url = ""
controleur = "controleurs/accueil.py"
template = "accueil.html"
```

fichier 7 (extrait)

```
body {
    background-color: #FFFFFF;
    box-sizing: border-box;
    margin: 0;
}
```

Exercice 1 Compréhension de l'architecture MVC

Le site est structuré selon l'architecture MVC. Il utilise le principe de routage en chargeant un fichier `routes.toml` et possède un fichier de configuration `init.py`. Les fichiers précédents correspondent au code de la page d'accueil, qui affiche un formulaire de recherche. Ignorez les `'@@@@'` pour l'instant.

1. En analysant les fichiers numérotés de 1 à 7, déduisez-en le nom de chacun de ces fichiers.
2. Décrivez le cheminement dans les fichiers quand on demande la page `index`.
3. On souhaite développer la page de vote pour une activité. Quelles modifications devez-vous apporter aux fichiers existants ? Créez ensuite la page, dont le contenu se compose d'un titre et du paragraphe *"Page*

disponible prochainement".

Exercice 2 Compréhension de code

1. Le code des fichiers 4, 5 et 7 est incomplet (au niveau des huit `'@@@@'`), à vous de le compléter !

Exercice 3 Écriture de code

1. Complétez la fonctionnalité permettant de voter pour une activité : le contrôleur et la vue (sous forme de template) sont fournis, mais pas les fonctions du modèle appelées par le contrôleur (*insert_vote, get_activites et get_semaines*).

controleurs/voter.py

```python
from model.model_pg import insert_vote, get_activites, get_semaines

if POST and 'bouton_valider' in POST:
  id_user = 1  # simplification
  id_activite = POST['activite'][0]
  sem_split = POST['semaine'][0].split('-')
  annee = sem_split[0]
  semaine = sem_split[1]
  result_insert = insert_vote(SESSION['CONNEXION'], id_user, id_activite, annee, semaine)
  if result_insert and result_insert > 0:
    REQUEST_VARS['message'] = "Le vote est enregistré !"
    REQUEST_VARS['message_class'] = "alert-success"
  else:
    REQUEST_VARS['message'] = "Erreur lors de l'enregistrement du vote."
    REQUEST_VARS['message_class'] = "alert-error"

REQUEST_VARS['activites'] = get_activites(SESSION['CONNEXION'])
REQUEST_VARS['semaines'] = get_semaines(SESSION['CONNEXION'], True)
```

templates/voter.html

```jinja
{% extends "base.html" %}

{% block main_content %}
<h1>Votez pour une activité</h1>

<form method="post" action="#">
  <label for="activite">Activité : </label>
  <select name="activite" id="activite">
    {% for act in REQUEST_VARS['activites'] %}
      <option value="{{ act[0] }}">{{ act[1] }}</option>
    {% endfor %}
  </select>
  <br/>
  <label for="semaine">Semaine : </label>
  <select name="semaine" id="semaine">
    {% for sem in REQUEST_VARS['semaines'] %}
      <option value="{{ sem[0] }}">{{ sem[0] }}</option>
    {% endfor %}
  </select>
  <br/>
  <input type="submit" name="bouton_valider" value="Voter"/>
</form>

{% include 'message.html' %}
{% endblock %}
```