Devoir/projet/TP fil rouge

2024-2025

Licence Informatique 3ème année

Alexandre Niveau

Devoir/projet/TP fil rouge

2024-2025

La moitié de la note pour le module « Programmation d'applications web avancées » est un DM/projet à faire en partie en TP. (L'autre moitié est un TP noté qui aura lieu le 5 décembre.)

Il est question de construire incrémentalement un site web conformément à l'architecture présentée en cours. Le résultat de ce « TP fil rouge » constituera votre DM/projet. Le travail correspondant à l'énoncé des TP est à faire en binôme. On vous demande également **un complément par étudiant·e**, à faire individuellement. (NB: si vous n'avez pas de binôme, il n'est pas attendu que vous rendiez deux compléments.)

Votre rendu fera l'objet d'un oral obligatoire très court (une dizaine de minutes), visant à nous assurer que vous avez compris ce que vous avez rendu. Chaque étudiant·e sera interrogé·e en particulier sur « son » complément.

Sauf cas exceptionnels, les binômes sont à former au sein des groupes de TP (notamment car les oraux sont organisés par groupe, et les corrections sont faites par les chargé·es de TP).

Complément en groupe

En premier lieu, vous devez compléter, toujours en binôme, le travail fait en TP, en mettant en place, de façon propre, l'utilisation d'une base de données MySQL pour stocker les animaux. Pour ce faire, vous pouvez suivre le déroulé de cet exercice.

NB: il n'est pas nécessaire d'avoir fait le complément en groupe avant de s'attaquer aux compléments individuels.

Compléments individuels

Voilà les deux compléments que chaque binôme doit réaliser. Attention, encore une fois, chaque complément doit être réalisé individuellement : il faut donc s'accorder sur

qui fait quoi au sein du binôme, et la répartition devra être explicitée dans le README sur le dépôt Git. Cependant il doit bien y avoir un seul rendu final, dans lequel les deux compléments devront être présents ensemble. NB: si vous n'avez pas de binôme, on attend un seul complément.

NB: le 2e complément est plus long à décrire mais pas plus long ou difficile que le premier. Les deux compléments se veulent équilibrés, et indépendants.

❦ [Complément 1] Upload

Ajouter la possibilité d'uploader une image de chaque animal. Cela nécessitera de modifier le modèle (ajout d'un attribut « chemin vers l'image »), la vue (formulaire de création), et d'intégrer proprement la gestion de l'upload au contrôleur, dans le respect des bonnes pratiques de sécurité.

❦ [Complément 2] Mise en place d'une mini-API

Mettre en place un autre script, `api.php`, qui va faire appel à un autre routeur qui devra reconnaître les URL suivantes :

- `api.php?collection=animaux` doit renvoyer une représentation JSON de la liste des noms des animaux, de la forme suivante :

  ```
  [
    {
      "id": 42,
      "nom": "Médor",
    },
    {
      "id": 83,
      "nom": "Félix",
    },
    {
      "id": 128,
      "nom": "Denver",
    },
  ]
  ```

  NB, en particulier, qu'on veut les identifiants et les noms, mais rien d'autre.

- `api.php?collection=animaux&id=42` (avec 42 remplacé par un identifiant d'animal) doit renvoyer une représentation JSON de l'animal d'identifiant 42, de la forme suivante :

  ```
  {
    "nom": "Médor",
    "espece": "chien",
  ```

```
    "age": 23
}
```

Ces représentations JSON doivent être générées par une nouvelle classe de vue (notre classe `View` est spécialisée pour générer du HTML, or ce n'est pas ce qu'on veut ici). On veut cependant utiliser les méthodes `showInformation` et `showList` du contrôleur sans les modifier.

Vous pouvez tester votre API avec cette page. NB: votre API doit retourner un document dont le content-type est identifié comme du JSON.

Barème indicatif

- TP 08 : 5 points
- TP 09 : 5 points
- Complément en groupe : 4 points
- Complément individuel : 6 points (note individuelle)

Vous pouvez avoir un bonus jusqu'à 2 points si vous implémentez (proprement) les URL avec PATH_INFO (partie optionnelle à la fin du TP08).

Contraintes techniques

Le non-respect des contraintes ci-dessous entraînera des pénalités pouvant aller jusqu'à annuler la note.

- Les pages générées devront être **valides HTML5** selon le validateur du W3C (« Validate by URI » devrait fonctionner).
- Le site devra être sécurisé contre les attaques de type XSS (cross-site scripting), comme expliqué en cours.
- Les contraintes de rendu ci-dessous doivent être respectées.

Rendu

Pour que votre travail soit considéré comme rendu, vous devez remplir les trois conditions suivantes, 1h avant votre créneau de soutenance :

- Vous devez avoir pushé tout votre travail sur le dépôt Git créé pour votre binôme sur le gitlab unicaen
- Votre site doit être déployé sur le serveur `dev-LOGIN.users.info.unicaen.fr` d'un des membres du binôme ; l'URL doit être présente dans le README sur le dépôt Git.
- Un·e des membres du binôme doit déposer une archive .zip contenant le répertoire exoMVCR sur ecampus, au plus tard une heure avant le début des oraux de votre groupe de TP. Cette archive fera foi en cas de doute. Le nom de

la personne ayant effectué le dépôt doit être précisé dans le README sur le dépôt Git.

❦ Précisions sur le dépôt Git

Les énoncés des TP 08 et 09 indiquaient de faire des commits à plusieurs endroits. On s'attend à retrouver ces commits dans l'historique de votre dépôt. Dans le cas contraire vous serez pénalisé·es. NB: si vous avez déjà suivi l'unité l'an passé, vous devez refaire l'exercice.

Par ailleurs, il faudra **au minimum** un commit par complément. N'oubliez pas de pusher vos modifications vers le dépôt distant, et de puller régulièrement.

Récapitulatif du contenu du README sur le dépôt Git :

- les noms, numéros étudiants et logins des membres du binôme, avec la répartition des compléments (qui a fait quoi)
- l'URL du site déployé sur le serveur d'un·e des membres
- le nom de l'étudiant·e qui a déposé le .zip sur ecampus

Planning des oraux

Les oraux auront lieu les jeudi et vendredi 5-6 décembre, planning à préciser. L'oral est obligatoire, les travaux remis sans soutenance seront évalués à 0.