Projet Médiathèque
Python, Django DB

SOMMAIRE :

1. 0 Présentation du projet

La médiathèque « Notre livre, notre média »

Vous a contacté afin de moderniser son système de gestion interne qui est pour l'instant géré à l'ancienne par des fiches papiers. L'objectif est d'avoir un système fonctionnel, sécurisé, utilisable par les bibliothécaires et également par le public.

1. 1 Les objectifs du projet

Objectifs:

La médiathèque propose à la consultation et à l'emprunt des livres, des Cds, des jeux de plateaux et des dvd. Tous ces supports sont ouverts à la consultation et à l'emprunt, sauf les jeux de plateau qui sont uniquement disponibles en consultation. Il y aura deux applications à déployer :

- Une application principale qui ne sera accessible qu'aux bibliothécaires.
- Une application de consultation accessible aux emprunteurs.

L'application bibliothécaire permettre:

- Créer un membre-emprunteur.
- Afficher la liste des membres.
- Mettre à jour un membre.
- Afficher la liste des médias.
- Créer un emprunt pour un média disponible.
- Ajouter un media.
- Rentrer un emprunt.

L'application bibliothécaire permettre :

L'application membre doit permettre uniquement d'afficher la liste de tous les médias.

1. 2 Étude et correction du code

```
def menu():
    print("menu")

if __name__ == '__main__':
    menu()


class livre():
    name = ""
    auteur = ""
    dateEmprunt = ""
    disponible = ""
    emprunteur = ""

class dvd():
    name = ""
        realisateur = ""
        dateEmprunt = ""
        disponible = ""
    emprunteur = ""


class cd():
    name = ""
    artiste = ""
    dateEmprunt = ""
    disponible = ""
    emprunteur = ""

class jeuDePlateau :
    name = ""
    createur = ""

class Emprunteur():
    name = ""
        bloque = ""


def menuBibliotheque() :
    print("c'est le menu de l'application des bibliothéquaire")




def menuMembre():
    print("c'est le menu de l'application des membres")
    print("affiche tout")
```

Correctif:

- Les fonctions de menu, menuBibliotheque, menuMembre seront supprimées et gérées par les views au sein des applications.

- Le script : **if __name__ == ' __main__ ' :**
menu()
sera supprimé et géré par defaut dans le dossier manage.py du projet.
if __name__ == ' __main__ ' :
main()

- Afin de respecter les conventions pour le bon fonctionnement du code en Python, il est nécessaire respecter l'indentation du code, ainsi les attributs "réalisateur", "dateEmprunt", "disponible" de la class dvd. Ainsi que l'attributs bloque de la class Emprunteur, on actuellement 4 espace en trop.

- Chaque nom de class qui doivent commencer par une majuscule, seule la class Emprunteur respect cette convention.

- Afin de répondre aux exigences du devoir et éviter les répétitions, nous utiliserons l'héritage de class. Ainsi une class mère **Media** regroupera l'ensemble des attributs communs quelle donnera en héritage aux class filles **Livre**, **Dvd**, **Cd.**

1. 3 Fonctionnalités

Afin de répondre au mieux au besoin du projet :

- Mise en place du environnement virtuel de Python et importation du framework Django

- Création du projet médiathèque qui contiens la page d'accueil, afin de rediriger l'utilisateur vers deux applications distingues :

- Application bibliothécaire contient les **models** des médias et les **formulaires,** ainsi que les **CRUD** permettant d'afficher et réaliser la liste, la création et la location des médias en fonction de sa catégorie.

- Application membre contient les **models** des membres et les **formulaires**, ainsi que les **CRUD** permettant d'afficher la liste des membres, de mettre à jour un membre ou de le supprimer.

1. 4 Stratégie de tests

Tests:

- Utilisation de Pytest, une librairie permettant d'écrire et de gérer les tests d'un programme Python. C'est un outil simple d'utilisation permettant de mettre en place tout type de tests.

- Pour installer Pytest, exécutez la commande suivante: **pip install -U pytest**

- Afin que les tests soient reconnus en tant que tels et soient exécutés par Pytest, il faut qu'ils se trouvent dans un fichier dont le nom est préfixé par **test_** , par exemple **test_models**.

- Dans le terminal, la commande > **Pytest** va lancer tous les tests contenus dans les fichiers test_* du projet.

1. 5 Exécution du programme

Exécution:

- Récupération repository GitHub :
  git clone https://github.com/DeschampsJulien/App_Mediatheque.git

- lancement du programme en local : **python manage.py runserver**

- Création d'un super utilisateur : **python manage.py createsuperuser**
  Login: **mediatheque**
  Mot de passe : **admin**