Rapport Médiathèque

---

1. Étude et correctifs du code fourni

Ce que j'ai gardé

J'ai conservé les idées principales du modèle métier qui sont :

- La distindéction entre les différents types de médias (Livre, CD, DVD, Jeux De Plateau)
- L'idée de membre/emprunteur
- L'idée de prêt, de disponibilité et de date d'emprunt

Ces élements mon servi de base pour concevoir les models (Membre, Media,Loan).

Ce que je n'ai pas gardé

Plusieurs parties du code initial que je n'ai pas retenu car elle n'etaient pas adaptées :

- Il n'yavait aucune gestion rélle des donées
- Le code était prévu uniquement pour le terminal (menus print)
- Il n'y avait pas de séparation claire entre les médias, membres et emprents
- Les régles métier n'étaient pas implémentées

Ces élements ont été entièrement refait pour permettre :

- Une application web avec des pages HTML
- Une base de données
- Des formulaires
- Une meilleure organisation du code

Ces éléments ont été **entièrement refaits** avec Django, qui permet :

- Une application web avec pages HTML
- Une base de données fiable
- Des formulaires sécurisés
- Une meilleure organisation du code (models, views, templates)

2. Mise en place des fonctionnalités demandées

- Gestion des médias
    - Création de médias avec title et media-type.

- Gestion des membres
    - Création de membres avec firstname, lastname, email.

- Gestion des emprunts
    - Association entre un membre et un media.
    - Enregistrement automatique de la date d'emprunt.

- Règles métiers
    - Les jeux ne sont pas empruntable mais disponible pour une consultation.
    - Les autres médias sont disponible et empruntable.

3. Stratégie de tests

Des tests unitaires ont été mis en place à l'aide du framework de tests Django. Chaque fonctionnalités principale (création de médias, modification, règles métier, gestion des membres et des emprunts) dispose d'au moins un test permettant de vérifier son bon fonctionnement. Ces test garantissent la fiabilité et la cohérence de l'application.

4. Base de données avec données de test

Médias

title	media_type	consultation_only	available
Test title	books	True	True
Test title 2	dvds	True	True
Test title 4	game	False	True

Membres

firstname	lastname	email
J-P	NOZA	jpnoza@hotmail.com
Jean	NOZA	jeannoza@hotmail.fr

Emprunts (`Loan`)

member	media	loan_date
Jean NOZA	Test title	Date du test auto

- Les données sont créées automatiquement par les tests et ne persistent pas en base de production
- Cela garantit un environnement propre, répétable et isolé.

5. Instructions pour exécuter le programme depuis n'importe quelle machine

1. **Télécharger le projet**
   - Cloner le dépôt ou copier tous les fichiers du projet sur la machine.
2. **Installer Python**
   - Le projet fonctionne avec Python 3.13+. Installer Python si nécessaire.
3. **Créer un environnement virtuel**
   - python -m venv .venv
   - source .venv/bin/activate # Linux/macOS
   - .venv\Scripts\activate # Windows
4. **Installer les dépendances**
   - pip install -r requirements.txt
5. **Appliquer les migrations**
   - python manage.py makemigrations
   - python manage.py migrate
6. **Exécuter le serveur**
   - python manage.py runserver
7. **Exécuter les tests**
   - python manage.py test

6. Conclusion

Le travail initial a été utile pour comprendre le besoin fonctionnel, mais il a fallu repenser complètement l'architecture pour passer d'une application console basique à une application web Django structurée et maintenable. Les concepts ont été conservés, mais l'implémentation a été améliorée pour respecter de bonnes pratiques de développement. Les tests unitaires assurent la fiabilité des fonctionnalités et des règles métiers.