Rapport de développement de l'application Médiathèque

Introduction

L'application Médiathèque est une application réalisée avec le framework Django qui permet la gestion des médias (livres, DVD, CD, jeux de plateau) et facilite le suivi des emprunts réalisés par les membres.
Elle permet aux bibliothécaires d'ajouter, consulter, et gérer les médias et aux membres d'emprunter des items disponibles.
Ce rapport détaille les correctifs appliqués au code initial, la mise en place des fonctionnalités, la stratégie de tests adoptée et les configurations de la base de données.

L’application bibliothécaire devra permettre de :

- 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 membre :

Elle doit permettre uniquement d’afficher la liste de tous les médias.

Les contraintes :

- Un membre ne peut pas avoir plus de 3 emprunts à la fois.
- Un emprunt doit être retourné au bout d’1 semaine.
- Un membre ayant un emprunt en retard ne peut plus emprunter.
- Les jeux de plateaux ne sont pas concernés par les emprunts.

Étude et correctifs du code fourni

Le code initial comportait plusieurs classes Python non fonctionnelles et manquait de structure pour une application web Django. Voici les correctifs principaux apportés :

1. Organisation des classes : Les classes Livre, DVD, CD, et JeuDePlateau ont été simplifiées et regroupées dans une classe parente Media, pour une gestion cohérente.
Les sous-catégories ont été définies via un champ type (par exemple : Livre, DVD, CD, etc.), ce qui permet de stocker divers types de médias dans une même table.

2. Ajout des attributs spécifiques : Chaque type de média a des attributs spécifiques (auteur pour les livres, réalisateur pour les DVD, artiste pour les CD, et fabricant pour les jeux de plateau).
Ces champs ont été ajoutés dans le modèle Media avec des options conditionnelles pour les rendre accessibles selon le type.

3. Correction des messages d'affichage : Certains messages de la page HTML s’affichaient de manière incorrecte, incluant des messages de statut dans le texte.
Nous avons structuré l’affichage des messages pour séparer les informations dynamiques et le contenu statique.

4. Ajout de l'attribut disponible : Un champ booléen disponible a été ajouté pour chaque média, permettant de suivre la disponibilité.
Cette mise à jour a nécessité une migration pour inclure ce champ dans la base de données.

5. Ajout de la gestion des emprunteurs : Une relation entre Membre et Media a été configurée pour suivre les emprunts.
Si un membre a des emprunts en retard, il ne peut plus emprunter jusqu’à la restitution des médias.
Cette logique est gérée au niveau du modèle.

Mise en place des fonctionnalités demandées

1. Gestion des Médias :

- Les bibliothécaires peuvent ajouter des médias en spécifiant le type (Livre, DVD, etc.), le titre, l'auteur/réalisateur/artiste/créateur, et la disponibilité.

Une liste des médias est affichée dans l'application, montrant leur disponibilité .

2. Emprunt et Gestion des Membres :

- Les membres peuvent emprunter des médias s'ils sont disponibles. Lorsqu'un média est emprunté, le champ disponible est mis à jour.

Une fonction de vérification de retard est en place. Si un membre a un retard dans un emprunt précédent, il est bloqué pour de futurs emprunts jusqu'à la restitution.

3. Interface utilisateur :

- Les pages HTML ont été simplifiées pour permettre une navigation facile. Des pages spécifiques pour la liste des médias, le formulaire d'emprunt, et les détails des membres ont été mises en place.

Les messages de succès et d'erreur s'affichent clairement, indiquant à l'utilisateur l'état de chaque opération.

4. Intégration de la base de données :

- La base de données a été configurée pour stocker les informations sur les membres, les médias, et les emprunts.

Une migration a été appliquée pour inclure les champs ajoutés (comme disponible, type, date_emprunt).

Stratégie de tests

1. Tests unitaires pour les modèles :

Tests de création et de sauvegarde de chaque type de média pour vérifier que tous les champs obligatoires sont bien sauvegardés.

Tests pour vérifier que la disponibilité d'un média est mise à jour correctement lors d'un emprunt ou d'un retour.

Tests pour la fonctionnalité de blocage d'un membre en cas de retard d'emprunt.

2. Tests fonctionnels :

Tests pour simuler un emprunt complet et vérifier l'évolution de l'état disponible du média.

Test de blocage pour s'assurer qu'un membre avec un emprunt en retard ne peut pas emprunter de nouveau.

Les tests sont automatisés avec pytest et incluent des cas valides et invalides pour s'assurer que l'application réagit correctement dans toutes les situations.

Pour chaque test, des données fictives ont été insérées dans la base de données MySQL afin de simuler des scénarios réalistes.

4. Base de Données avec Données Test (MySQL via PhpMyAdmin)

La base de données a été configurée avec les tables suivantes dans PhpMyAdmin :

- Table media : Contient les informations sur chaque média (ID, titre, type, disponible, etc.).
- Table emprunteur : Contient les informations sur les membres emprunteurs, y compris le statut de blocage.
- Table emprunt : Enregistre les emprunts effectués, avec une relation entre les emprunteurs et les médias.

Des données de test ont été insérées dans chacune des tables pour vérifier le bon fonctionnement des fonctionnalités. Voici quelques exemples d'entrées dans la table media :

Instructions pour Lancer le Projet de Médiathèque

Prérequis

Avant de commencer, assurez-vous d'avoir les éléments suivants installés sur votre machine :

- Python (version 3.6 ou supérieure)
- Git pour cloner le projet
- MySQL
- pip pour gérer les packages Python
- PhpMyAdmin

Étapes à Suivre

1. Cloner le Projet depuis GitHub :

Ouvrez votre terminal (Invite de commande sur Windows ou Terminal sur macOS/Linux).

Clonez le dépôt Git avec la commande suivante :

- git clone https://github.com/kilianaBldr/Mediaheque_app.git

Accédez au dossier du projet cloné :

- cd Mediaheque_app

2. Créer un Environnement Virtuel :

Créez un environnement virtuel pour isoler les dépendances de votre projet :

- python -m venv mediatheque_env

3. Activer l'Environnement Virtuel :

Activez l'environnement virtuel :

Sur Windows :

- mediatheque_env\Scripts\activate

Sur macOS/Linux :

- source mediatheque_env/bin/activate

4. Installer les Dépendances :

installez Django et le client MySQL :

- pip install django
- pip install mysqlclient

5. Importer le Script SQL :

Assurez-vous que le script SQL (par exemple, script.sql) présent sur Github

Ouvrez le terminal MySQL et connectez-vous :

- Importez le script SQL dans votre base de données :

7. Appliquer les Migrations :

Exécuter les commandes suivantes :

- python manage.py makemigrations
- python manage.py migrate

8. Démarrer le Serveur de Développement :

Lancez le serveur de développement Django avec la commande :

- python manage.py runserver

9. Accéder à l'Application :

Ouvrez un navigateur et accédez à http://127.0.0.1:8000/ pour voir votre application en cours d'exécution.

Compte Administrateur Django

Username : admin
Email : admin@exemple.com
Mot de passe : bibliothecaire