

Réseau social sur les livres

[Introduction](#)

[Données](#)

[Fonctionnalités](#)

[Interface utilisateur \(UI\)](#)

[Documentation](#)

[Technologies à utiliser](#)

[Échéancier et rendu](#)

[Critères d'évaluation](#)

[Ressources](#)

▼ Introduction

L'objectif de ce projet est de créer une plateforme qui permettra aux utilisateurs de partager, de discuter et de découvrir des livres en fonction de leurs intérêts. Cette application aura la particularité d'intégrer l'API SRU du catalogue général de la Bibliothèque Nationale de France (BNF).

▼ Données

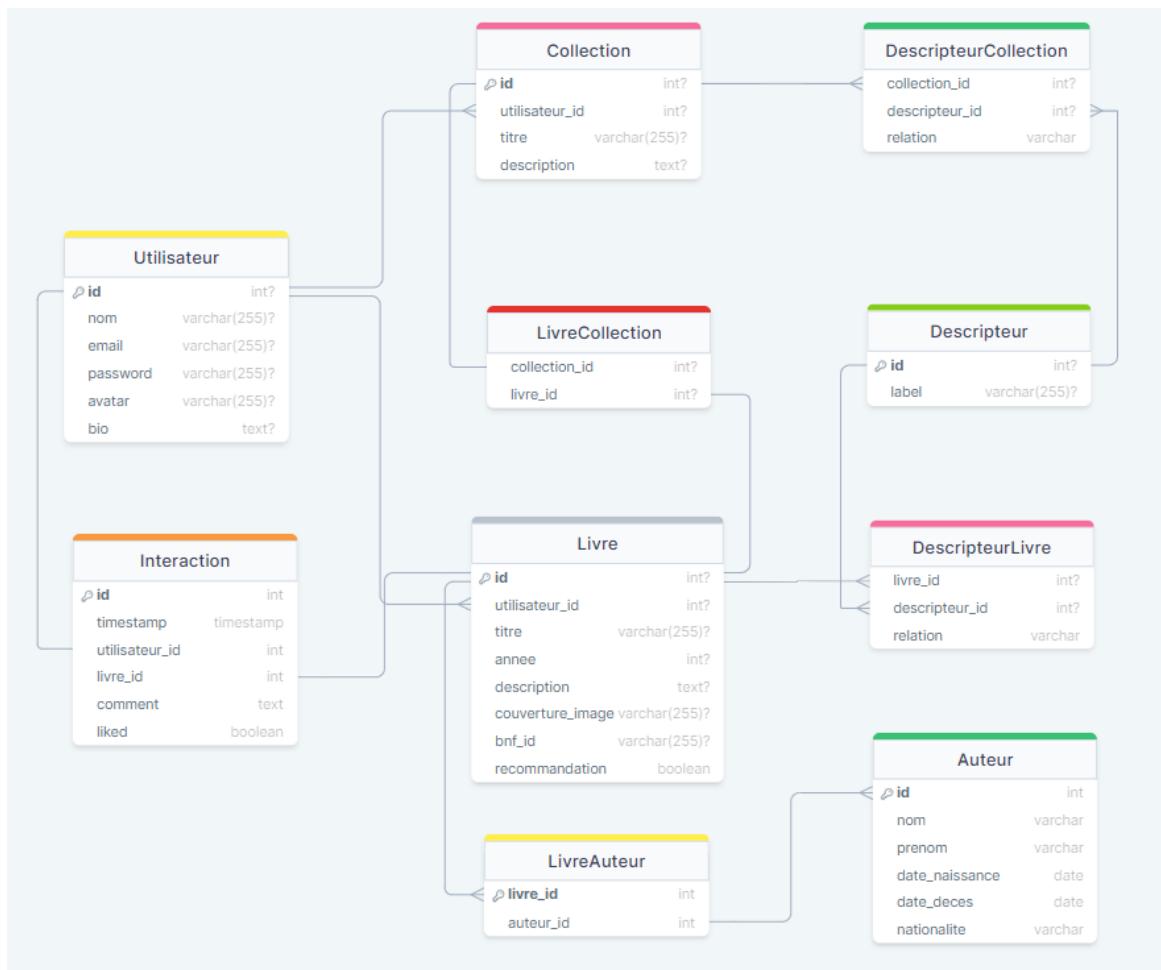
L'application comportera 2 sources de données:

- une base de données interne pour gérer les utilisateurs, leurs interactions et leurs livres et collections. Le choix du système de base de données relationnelle est au choix (MySQL, MariaDB, SQLite, PostgreSQL,

OracleSQL, etc.) mais doit être documenté de son installation à la création des tables et leur éventuel pré-remplissage

- l'[API SRU du catalogue général de la BNF](#) pour y effectuer les recherches de livres et récupérer les données

Pour vous aider dans la conception de la base de données, voici un modèle de données non exhaustif qui peut vous donner les grandes lignes du modèle à mettre en place



▼ Fonctionnalités

L'application devra comprendre les Fonctionnalités requises, ainsi qu'au minimum deux autres fonctionnalités (voir la partie sur les Fonctionnalités imaginables).

▼ Fonctionnalités requises

1. Inscription, Connexion et Déconnexion des utilisateurs

2. Profil Utilisateur :

- Chaque utilisateur aura un profil contenant des informations telles que le nom, le mail et une courte biographie.
- Le mot de passe doit pouvoir être changé

3. Ajout et Modification de Livres avec intégration de l'API SRU

Catalogue général de la BNF :

- Les utilisateurs pourront ajouter des livres à leur profil en utilisant l'API de la BNF.
- Les détails des livres seront automatiquement récupérés à partir des API et associés à l'entrée de l'utilisateur: les données récupérées seront alors stockées dans la base de l'application pour permettre à l'utilisateur de les modifier si besoin.
- Les utilisateurs pourront ajouter des critiques et des évaluations pour chaque œuvre de son profil ou de celui d'un autre utilisateur.

4. Crédit de collections personnelles

- L'utilisateur pourra ranger des livres dans des collections

5. Recherche et Découverte :

- Les utilisateurs pourront rechercher des livres en utilisant des filtres tels que le genre, l'auteur, l'année, des mots du titre, etc.
- Chaque résultat de recherche devra pouvoir être commenté ou mis en favoris.

6. Relations Utilisateurs :

- Les utilisateurs pourront mettre en évidence des livres sur leur profil.
- Ils pourront interagir avec les favoris d'autres utilisateurs en laissant des commentaires et des "j'aime", voire en leur attribuant eux-mêmes une évaluation.

7. Intégration de Médias :

- Les utilisateurs pourront ajouter des images de couverture et des images liées aux livres si le document est présent sur Gallica.
- Ajout d'une photo de profil (via une image par un lien, ou via un fichier téléchargé)

▼ Fonctionnalités imaginables

Ou toute autre idée à votre souhait

1. Connexion

- Création d'un nouveau mot de passe en cas de perte/d'oubli grâce à un lien temporaire envoyé par mail

2. Tableau de Bord :

- Chaque utilisateur aura un tableau de bord où il pourra voir des statistiques sur ses collections.

3. Système de Notifications :

- Les utilisateurs recevront des notifications pour les interactions sur des livres de son profil.

▼ Interface utilisateur (UI)

- Le choix de la mise en page, des styles, des couleurs, etc., est libre. La priorité doit être une interface fonctionnelle; un minimum de mise en page est attendu mais il est inutile de passer plus de temps sur le style que sur les fonctionnalités.

▼ Documentation

- Le code devra être correctement commenté comme précisé dans le cours : [Bonnes pratiques - commentaires](#)
- Le dépôt Git de rendu devra comprendre un *README.md* à sa racine conforme aux bonnes pratiques du cours : [Bonnes pratiques - Readme](#)
- Si le sujet nécessite une base de données, voir [Bonnes pratiques - bdd:](#)
 - indiquer précisément comment installer le SGBDR sur mon poste si ce n'est pas SQLite
 - fournir :
 - soit le fichier de base de données pour SQLite
 - soit les scripts de création des tables, des schémas éventuels, des users, etc., ainsi qu'un dump des données si un pré-remplissage de la base est nécessaire

Dans tous les cas, fournir le modèle relationnel de données (format .png, .pdf, etc.)

▼ Technologies à utiliser

- Python, entre autres librairies:
 - Flask
 - Flask-Login
 - Requests
- Base de données SQL (SGBDR au choix)
- HTML, CSS, JavaScript
- Versioning Git pour le développement en équipe

▼ Échéancier et rendu

Échéance	Etape
Cours du 4 décembre 2023	- Groupes de 3 étudiants formés - Sujet choisi
Entre le 4 décembre 2023 et le 15 ou 22 janvier 2024	- Conception de la maquette - Réflexion sur les différentes briques à mettre en place et leurs interactions - Conception de la base de données (si nécessaire) - Etude de la documentation des APIs si besoin
15 ou 22 janvier 2024	Revue de projet pré-développement par groupe (avant ou après les cours de Flask): dans les grandes lignes, ordre de développement des différentes briques
Entre mi-janvier et le 31 mars 2024	Développement de l'application
Vers fin février	Revue de projet par groupe (à distance): cas de blocage, questions diverses, etc.
Dimanche 31 mars 2024	Rendu définitif

- Le rendu se présentera sous la forme d'un **dépôt Github** public uniquement: je ne regarde pas les commit ni les personnes ayant fait telle ou telle ligne de code. En revanche, il est presque **indispensable d'utiliser git** et le développement par branches (une branche par étudiant) mergées de temps

en temps pour récupérer le travail des autres. Les sujets sont conçus pour que trois personnes puissent développer des briques différentes simultanément

- Le rendu devra être conforme aux **bonnes pratiques** listées dans le document ci-après: Bonnes pratiques

▼ Critères d'évaluation

La note finale est composée de trois parties:

- Bonnes pratiques de développement

Attendu	Points
Lancement de l'application sans erreur et conforme à la description du <i>README.md</i>	1
Gestion des erreurs	1
Commentaires de code	1
Structure applicative logique (nommages, rangement, variabilisation)	1
Chemins absous bannis	0.5
Liens internes correctement spécifiés	0.5
	5

- Application

Attendu	Points
<u>Sources de données</u> - Base de données relationnelle: branchement, modèles de données - API: lecture, parsing des réponses	/2
<u>Gestion des utilisateurs</u> - Inscription, connexion, déconnexion - Protection de pages	/1
<u>Front-end</u> - Structuration des fichiers de templates - Jinja2	/1
<u>Interactions avec les données</u> - Filtres de recherche - Pré-visualisation de documents depuis la réponse de l'/des API(s) - Création de collections - Ajout de documents, modification et suppression possibles	/3
<u>Interactions avec les utilisateurs</u> - Commentaires et évaluations de documents - Consultation de profils	/3
	10

- Fonctionnalités supplémentaires

Attendu	Points
Optimisation du code (non duplication notamment)	1
Fonctionnalité supplémentaire 1	2
Fonctionnalité supplémentaire 2	2
	5

▼ Ressources

- Documentation de l'API SRU du catalogue général de la BNF:
<https://api.bnf.fr/api-sru-catalogue-general>
- Documentation de l'API document de Gallica, et notamment de la manière de récupérer l'image d'une couverture : https://api.bnf.fr/fr/api-document-de-gallica#scroll-nav_7
- Des exemples de projets Flask intégrant des interactions entre utilisateurs, pour regarder la structure du code, le modèle de données, l'écriture des différents objets, etc. :
 - Le célèbre tuto Flask de Miguel Grinberg et le projet microblog :
<https://github.com/miguelgrinberg/microblog/tree/2023>
 - Un forum natif Python et Flask : <https://github.com/flaskbb>
 - Un forum créé en évaluation Flask il y a 3 ans:
<https://github.com/Chartes-TNAH/ForumTNAH>
- Réinitialisation de mot de passe : <https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-x-email-support>