

# Projet Services Web - EPSI Bachelor 3 DevOps

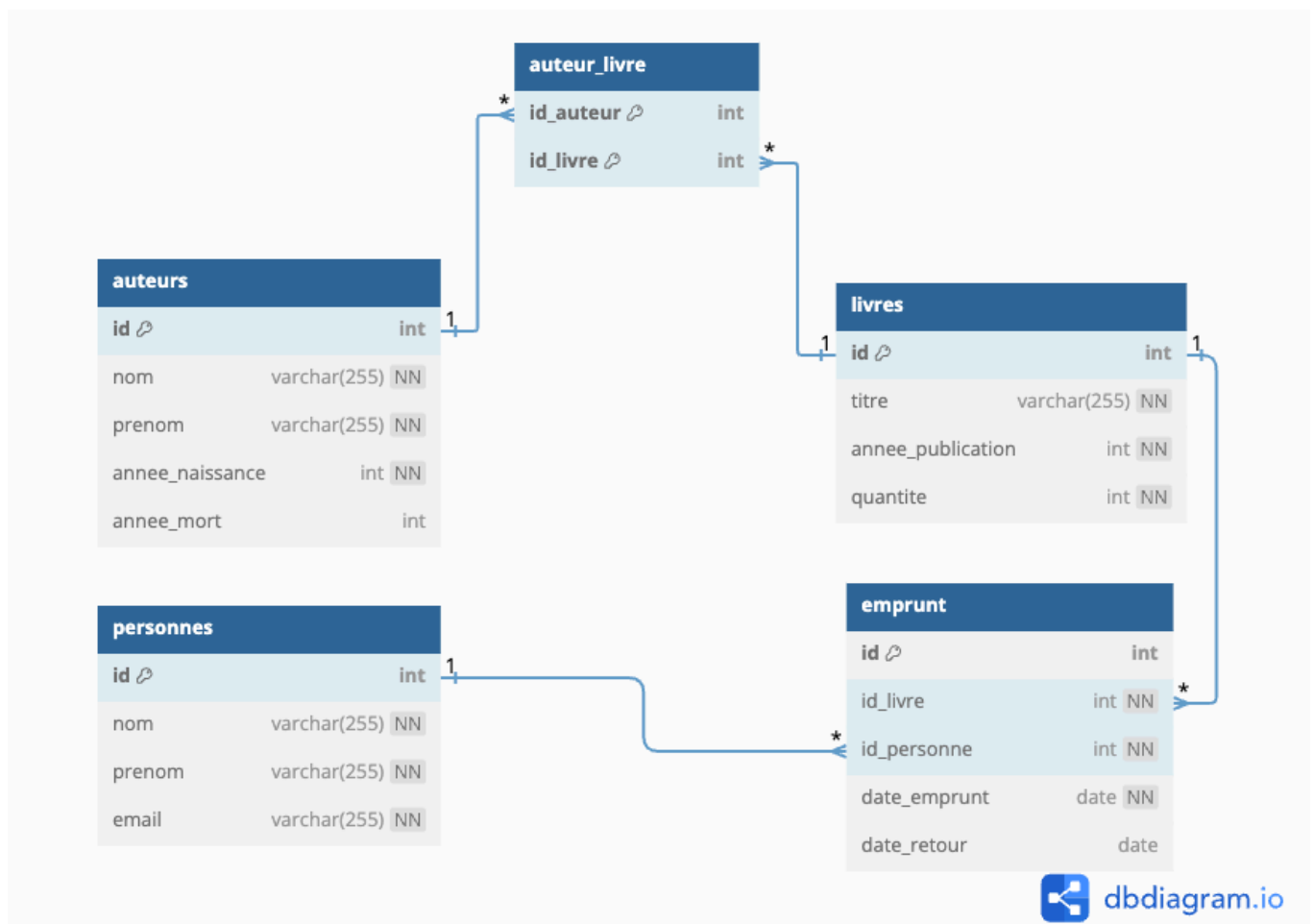
## I. Objectif

Construire une API RESTful en Node.js qui simule le fonctionnement d'une petite bibliothèque. Les utilisateurs peuvent ajouter des livres, emprunter des livres, et les retourner. L'API doit également inclure un mécanisme d'authentification par API Key et gérer les modifications concurrentes.

Exemple : [https://github.com/3rgo/node\\_api](https://github.com/3rgo/node_api)

## II. Modèle de Données

Votre API devra être capable de gérer une Base de Données Sqlite, avec le modèle de données suivant :












Les instructions SQL de création de la BDD sont fournies, ainsi qu'un jeu de données de départ. Tout autre champ pouvant être nécessaire au bon fonctionnement de l'API est libre dans son nommage.

### III. Consignes

Votre API devra respecter les consignes suivantes :

- Pouvoir recevoir et traiter du JSON sur les points d'entrée
- Renvoyer des ressources au format JSON
- Écouter sur le port 8000 selon le format d'URL "http://localhost:8000/api/..."
- Les réponses devront utiliser les codes de statut HTTP adaptés
- Être intégralement sécurisée via l'API Key "8f94826adab8ffebbeadb4f9e161b2dc".
- Utiliser un système d'ETag pour empêcher les modifications concurrentes (sans If-None-Match)

Les points d'entrée suivants devront être réalisés (les parties entre accolades sont des paramètres d'URL) :

Gestion des livres		
Méthode	URL	Description
GET	/livre	Retourne la liste des livres avec les informations des auteurs
GET	/livre/{id}	Retourne la fiche du livre portant l'ID indiquée, avec les informations des auteurs associés
POST	/livre	Crée le livre selon les informations du corps de la requête  Les auteurs sont fournis par leurs IDs  La quantité en stock est initialisée à 1 si elle n'est pas fournie  Retourne une erreur si un auteur n'existe pas
PUT	/livre/{id}	Modifie le livre selon les informations du corps de la requête  Les auteurs sont fournis par leurs IDs  La quantité en stock n'est pas modifiable  Retourne une erreur si un auteur n'existe pas
GET	/livre/{id}/quantite	Retourne la quantité totale et la quantité disponible pour le livre  Quantité disponible = quantité totale - nombre d'emprunts en cours
PUT	/livre/{id}/quantite	Modifie la quantité totale pour le livre  Retourne une erreur si la nouvelle quantité est inférieure au nombre d'emprunts actuellement en cours
DELETE	/livre/{id}	Supprime le livre  Retourne une erreur si des emprunts sont en cours

<u>Gestion des auteurs</u>		
Méthode	URL	Description
GET	/auteur	Retourne la liste des auteurs
GET	/auteur/{id}	Retourne la fiche de l'auteur portant l'ID indiquée
POST	/auteur	Crée l'auteur selon les informations du corps de la requête
PUT	/auteur/{id}	Modifie l'auteur selon les informations du corps de la requête
DELETE	/auteur/{id}	Supprime l'auteur ⚠ Retourne une erreur si l'auteur est utilisé par un ou plusieurs livres

<u>Gestion des emprunts</u>		
Méthode	URL	Description
POST	/emprunt	Crée un emprunt selon les informations du corps de la requête <i>i</i> Le livre est choisi par son ID <i>i</i> La date d'emprunt est remplie automatiquement à la date du jour <i>i</i> Les informations de l'emprunteur permettent de le créer dans la table personnes s'il n'existe pas ou de le modifier s'il existe déjà (identification via l'email) ⚠ Retourne une erreur si le le livre n'est pas empruntable (quantité disponible = zéro)
PUT	/emprunt/{id}	Modifie l'emprunt (remplis la date de retour)

<u>Recherche</u>		
Méthode	URL	Description
GET	/recherche/{mots}	Recherche des livres selon les mots fournis parmi le titre et le nom/prénom de l'auteur <i>i</i> Les résultats sont classés par taux de correspondance (à taux de correspondance égal, l'ordre n'a pas d'importance) <i>Exemple : la recherche "hugo misérables" retournera le livre "Les Misérables" de Victor Hugo, puis l'ensemble des livres de Victor Hugo (peu importe leur ordre)</i>

## IV. Rendu

Ce travail sera à réaliser seul ou en binôme (noms du binôme à indiquer sur 360 Learning). Le rendu se fera sous la forme d'un dépôt GitHub/GitLab public dont l'URL sera communiquée via **360 Learning**.

## V. Critères de notation

Vous serez évalués sur un total de 20 points, selon le barème suivant :

Réalisation	Valeur en points
Mise en place d'une API en NodeJS + ExpressJS écoutant sur le port 8000	1
Sécurisation via API Key	1
Validation des modifications via ETag	5
14 points d'entrée	14
Recherche	4
<b>Total</b>	<b>25</b>