Modélisation Complète de la Base de Données Cloud Firestore

Application Mobile **Biblio Fac**

Projet de Programmation Mobile – L4 LMD

11 février 2026

Table des matières

1 Introduction

Ce document présente la modélisation complète de la base de données Cloud Firestore de l'application mobile **Biblio Fac**.

L'objectif de cette modélisation est de :

— Structurer les données de manière logique et normalisée

— Assurer la cohérence entre Firebase Authentication et Firestore

— Faciliter la gestion des rôles (Étudiant / Administrateur)

— Garantir la sécurité via des règles Firestore adaptées

L'application repose sur quatre collections principales :

— **users**

— **books**

— **loans**

— **reservations**

2 Collection `users`

2.1 Rôle de la collection

La collection `users` contient les informations académiques et personnelles des utilisateurs authentifiés via Firebase Authentication.

L'identifiant du document correspond directement au **UID généré par Firebase Auth**, ce qui garantit la cohérence entre le système d'authentification et la base de données.

Cette collection permet également la gestion des rôles (Étudiant / Administrateur) et facilite le suivi administratif des étudiants grâce à leurs informations universitaires.

2.2 Structure d'un document

```
users (collection)
   |
    uid (document)
        - fullName: string
        - email: string
        - matricule: string
        - faculty: string
        - promotion: string
        - phoneNumber: string
        - address: string
        - role: string ("student" | "admin")
        - createdAt: timestamp
        - lastLogin: timestamp
        - isActive: boolean
```

2.3 Description des champs

— **fullName** : Nom complet de l'utilisateur.

— **email** : Adresse email utilisée pour l'authentification.

— **matricule** : Numéro académique unique de l'étudiant. Champ obligatoire pour les étudiants. Permet l'identification officielle à l'université.

— **faculty** : Faculté ou département d'appartenance (exemple : Informatique, Médecine, Droit).

— **promotion** : Niveau académique actuel (exemple : L1 LMD, L3 LMD, M1, M2).

— **phoneNumber** : Numéro de téléphone de l'utilisateur.

— **address** : Adresse de résidence de l'utilisateur.

— **role** : Définit le rôle de l'utilisateur (*student* ou *admin*).

— **createdAt** : Date de création du compte.

— **lastLogin** : Date et heure de la dernière connexion.

— **isActive** : Indique si le compte est actif ou désactivé.

3 Collection `books`

3.1 Rôle de la collection

La collection `books` représente le catalogue complet des livres disponibles dans la bibliothèque universitaire.

3.2 Structure d'un document

```
books
   |
    bookId
        - title: string
        - author: string
        - isbn: string
        - description: string
        - coverUrl: string
        - category: string
        - totalCopies: number
        - availableCopies: number
        - publishedDate: string
        - createdAt: timestamp
        - updatedAt: timestamp
```

3.3 Description des champs

— **title** : Titre du livre.

— **author** : Auteur du livre.

— **isbn** : Identifiant ISBN unique.

— **description** : Résumé du livre.
— **coverUrl** : URL de la couverture (Google Books API).
— **category** : Catégorie académique.
— **totalCopies** : Nombre total d'exemplaires physiques.
— **availableCopies** : Nombre d'exemplaires disponibles.
— **publishedDate** : Date de publication.
— **createdAt / updatedAt** : Gestion des modifications.

4 Collection `loans`

4.1 Rôle de la collection

La collection `loans` permet de suivre tous les emprunts effectués par les étudiants.

4.2 Structure

```
loans
   |
    loanId
        - userId: string
        - bookId: string
        - loanDate: timestamp
        - dueDate: timestamp
        - returnDate: timestamp (nullable)
        - status: string
          ("pending" | "approved" | "returned" | "rejected")
```

4.3 Description

— **userId** : Référence vers l'utilisateur.
— **bookId** : Référence vers le livre.
— **loanDate** : Date de demande.
— **dueDate** : Date limite de retour.
— **returnDate** : Date effective de retour.
— **status** : Statut de l'emprunt.

5 Collection `reservations`

5.1 Rôle

Permet aux étudiants de réserver un livre lorsqu'il n'est pas disponible.

5.2 Structure

```
reservations
   |
    reservationId
        - userId: string
        - bookId: string
        - reservationDate: timestamp
        - status: string
          ("active" | "cancelled" | "fulfilled")
```

6 Relations entre les collections

— Un utilisateur peut avoir plusieurs emprunts.

— Un utilisateur peut avoir plusieurs réservations.

— Un livre peut être associé à plusieurs emprunts.

— Un livre peut être associé à plusieurs réservations.

Les relations sont établies via les identifiants `userId` et `bookId`.

7 Mise en pratique dans Firestore

7.1 Création des collections

1. Aller dans Firebase Console
2. Ouvrir Firestore Database
3. Cliquer sur **Start Collection**
4. Créer successivement : users, books, loans, reservations

7.2 Configuration des règles de sécurité

La règle actuelle est temporaire et dangereuse :

```
allow read, write: if request.time < timestamp.date(2026, 3, 13);
```

Cette règle autorise tout accès jusqu'à la date indiquée.

7.3 Règles sécurisées recommandées

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {

    function isAuthenticated() {
      return request.auth != null;
    }
```

```
    function isAdmin() {
      return get(/databases/$(database)/documents/users/$(request.auth.uid))
        .data.role == "admin";
    }

    match /users/{userId} {
      allow read, write: if request.auth.uid == userId;
    }

    match /books/{bookId} {
      allow read: if isAuthenticated();
      allow write: if isAdmin();
    }

    match /loans/{loanId} {
      allow create: if isAuthenticated();
      allow read: if isAuthenticated();
      allow update: if isAdmin();
    }

    match /reservations/{reservationId} {
      allow create: if isAuthenticated();
      allow read: if isAuthenticated();
      allow update: if isAuthenticated();
    }
  }
}
```

Ces règles permettent :

— Accès restreint aux utilisateurs connectés

— Modification des livres uniquement par les administrateurs

— Protection des données personnelles

8 Conclusion

La modélisation Firestore de **Biblio Fac** repose sur une structure claire, normalisée et sécurisée.

Elle permet :

— Une séparation claire des responsabilités

— Une gestion efficace des rôles

— Une évolutivité future

— Une conformité aux exigences académiques