Fiche de Documentation Technique – Application Bibliothèque

Informations Générales

Nom du projet : Application Bibliothèque
Type : Application mobile multiplateforme
Technologie principale : Flutter (Dart)
Public cible : Bibliothécaires, administrateurs de bibliothèque

Objectif du projet :
Développer une application mobile permettant de gérer efficacement un catalogue de livres. L'application offre aux utilisateurs autorisés (principalement les bibliothécaires) la possibilité de créer, modifier, supprimer et consulter des livres et auteurs. Elle vise à améliorer la gestion des ressources d'une bibliothèque à travers une interface intuitive et responsive.

Architecture de l'Application

Modèle architectural : MVVM (Model-View-ViewModel)

Description des couches :

- **Model** : Contient les classes de données et entités métier, telles que Livre et Auteur. Ces classes incluent également la sérialisation/désérialisation pour interagir avec Firebase.
- **View** : Ensemble des interfaces utilisateurs, codées avec Flutter. Exemple : page de liste de livres, formulaire de création d'auteur, page de détail.
- **ViewModel** : Gère la logique métier et les états. Il agit comme intermédiaire entre le modèle et la vue. Exemple : LivreViewModel encapsule la logique de création, validation et mise à jour des livres.

Avantages de MVVM :

- Facilite la maintenance
- Testabilité améliorée
- Séparation claire des responsabilités
- Intégration facilitée avec des outils de gestion d'état (comme Provider ou Riverpod)

Technologies et Outils

Élément	Détail
Langage	Dart
Framework	Flutter
Base de données	Firebase Firestore (temps réel + NoSQL)
Authentification	Firebase Authentication (Email/Password)
IDE	Android Studio / VS Code
Contrôle de version	Git (GitHub ou GitLab recommandé)

Fonctionnalités Clés

- ◈ **Livres**
  - Ajout, modification, suppression
  - Visualisation des détails d'un livre
  - Association d'un livre à un auteur
- ◈ **Auteurs**
  - Gestion complète des auteurs
  - Lien automatique entre livres et auteurs
- ◈ **Interface Utilisateur**
  - Affichage sous forme de listes avec filtres
  - Formulaires avec validation et messages d'erreur clairs
- ◈ **Synchronisation temps réel**
  - Les modifications sont immédiatement visibles via Firebase

---

Gestion des Droits et Sécurité

- **Authentification** : Firebase Authentication (Email + mot de passe)
- **Rôles utilisateurs** :
  - **Administrateur** : Accès complet à toutes les fonctionnalités
  - **Utilisateur simple** : Accès en lecture seule

- **Sécurité des données** :
    - Règles de sécurité Firebase configurées pour restreindre l'accès en fonction des rôles
    - Validation côté client ET côté serveur

Tests

- **Type de tests effectués** :
    - Tests manuels sur simulateur Android et appareil physique
    - Contrôle de flux et validation via console Firebase
    - Tests d’interface utilisateur de base avec flutter_test
- **Tests à prévoir (recommandé)** :
    - Tests unitaires sur ViewModel
    - Tests d’intégration pour les interactions avec Firebase
    - Tests end-to-end (e2e) avec Flutter Driver ou integration_test

---

Déploiement

- **Environnement de test** : Android Emulator, smartphones Android physiques
- **Déploiement cible** (prévu) :
    - Android (Play Store)
    - iOS (App Store – en cours d’étude)
- **Processus de déploiement** :
    - Compilation via flutter build apk ou flutter build appbundle
    - Publication via Google Play Console
    - Utilisation de Firebase App Distribution (optionnelle)

Annexes

- Diagrammes de classes (modèle de données)
- Schéma MVVM de l’application
- Liste des packages Flutter utilisés

- Journal de version (changelog)
- Plan de tests