Projet FoodScan

SEGERS Jules

APPIA Assadou

EL BADAOUI Bilel

BENFARES Myriam

architecture logicielle de la solution complète

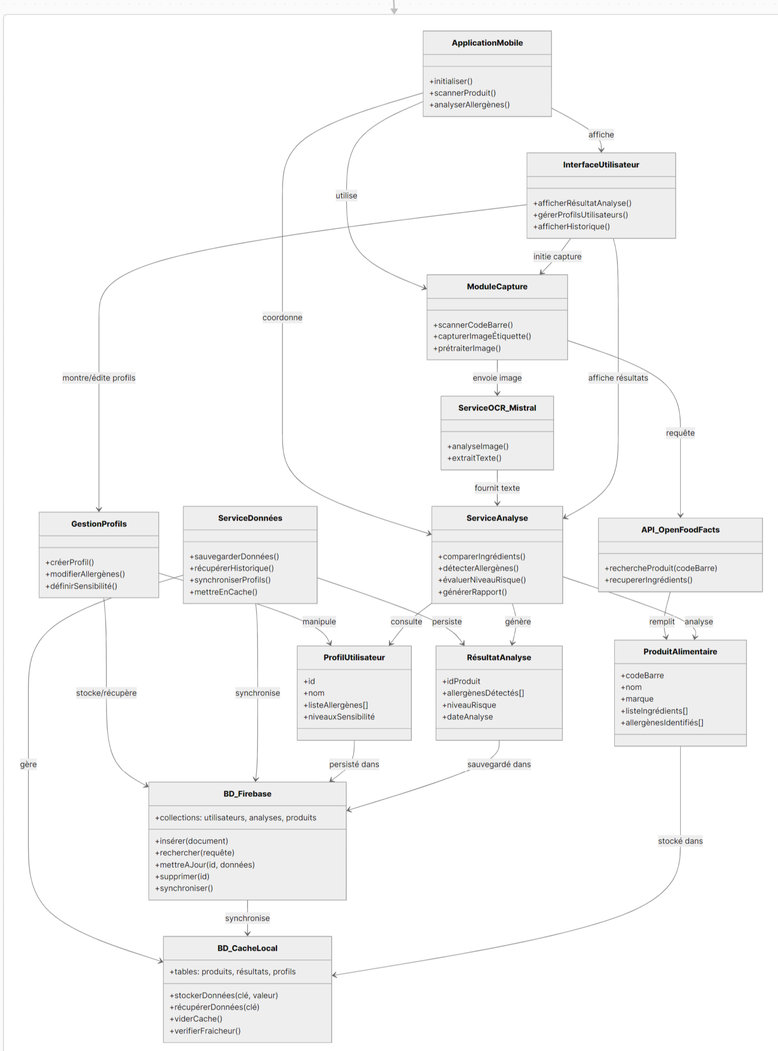


Tableau des fonctionnalités par composant

Composant	Fonctionnalités	Description
ApplicationMobile	initialiser()	Démarre l'application et initialise les composants nécessaires
	scannerProduit()	Lance le processus de scan d'un produit
	analyserAllergènes()	Coordonne le processus d'analyse des allergènes
InterfaceUtilisateur	afficherRésultatAnalyse()	Présente les résultats d'analyse à l'utilisateur
	gérerProfilsUtilisateurs()	Interface de gestion des profils utilisateurs
	afficherHistorique()	Affiche l'historique des scans et analyses
ModuleCapture	scannerCodeBarre()	Capture et décode les codes-barres de produits
	capturerImageÉtiquette()	Prend des photos d'étiquettes pour OCR
	prétraiterImage()	Optimise les images avant envoi à l'OCR
ServiceAnalyse	compareIngrédients()	Compare les ingrédients aux allergènes connus
	détecterAllergènes()	Identifie les allergènes potentiels
	évaluerNiveauRisque()	Évalue le niveau de risque pour l'utilisateur
	générerRapport()	Produit un rapport complet d'analyse
GestionProfils	créerProfil()	Crée un nouveau profil d'utilisateur
	modifierAllergènes()	Gère la liste des allergènes d'un profil
	définirSensibilité()	Configure les niveaux de sensibilité aux allergènes
BD_Firebase	insérer(document)	Ajoute des données dans la base cloud
	rechercher(requête)	Effectue des requêtes sur les collections
	mettreAJour(id, données)	Met à jour des documents existants
	supprimer(id)	Supprime des données
	synchroniser()	Synchronise les données cloud/local
BD_CacheLocal	stockerDonnées(clé, valeur)	Sauvegarde des données dans le cache local
	récupérerDonnées(clé)	Récupère des données du cache
	viderCache()	Nettoie le cache local
API_OpenFoodFacts	rechercheProduit(codeBarre)	Interroge l'API pour obtenir un produit
	recupererIngrédients()	Récupère la liste des ingrédients
ServiceOCR_Mistral	analyseImage()	Analyse une image d'étiquette
	extraitTexte()	Extrait le texte des ingrédients
ServiceDonnées	sauvegarderDonnées()	Persiste les données utilisateurs
	récupérerHistorique()	Récupère l'historique des analyses
	synchroniserProfils()	Synchronise les profils entre appareils
	mettreEnCache()	Met en cache les données fréquemment utilisées

Flux d'interactions principaux

Flux de scan par code-barres:

1. L'utilisateur initie un scan via **InterfaceUtilisateur**
2. **ModuleCapture** capture le code-barres
3. **ModuleCapture** envoie une requête à **API_OpenFoodFacts**
4. Les données du produit sont formatées en objet **ProduitAlimentaire**
5. **ServiceAnalyse** compare les ingrédients avec le **ProfilUtilisateur**
6. **ServiceAnalyse** génère un **RésultatAnalyse**
7. **InterfaceUtilisateur** affiche les résultats
8. **ServiceDonnées** sauvegarde l'analyse dans **BD_Firebase** et **BD_CacheLocal**

Flux d'analyse par OCR:

1. L'utilisateur capture une image via **InterfaceUtilisateur** et **ModuleCapture**
2. L'image est envoyée au **ServiceOCR_Mistral**
3. Le texte extrait est utilisé pour créer un objet **ProduitAlimentaire**
4. Le reste du flux suit le même chemin que le scan par code-barres

1. Composants Frontend (Flutter)

Ces composants gèrent l'interface utilisateur et les interactions directes avec l'utilisateur:

- **ApplicationMobile**: Composant principal qui sert de point d'entrée à l'application. Il coordonne les autres composants et orchestre le flux général des opérations.
- **InterfaceUtilisateur**: Gère tous les éléments d'interface visibles par l'utilisateur, l'affichage des résultats et la navigation.
- **ModuleCapture**: Responsable de toutes les fonctionnalités de capture par l'appareil photo de l'utilisateur (code-barres, photos) pour l'identification des produits.

2. Services Métier

Ces composants implémentent la logique fonctionnelle centrale de l'application:

- **ServiceAnalyse**: Partie analytique de l'application qui traite les données des produits pour identifier les allergènes.
- **GestionProfils**: Gère les profils utilisateurs et leurs préférences d'allergènes.

3. Modèles de Données

Ces composants définissent les structures de données principales manipulées par l'application:

- **ProfilUtilisateur**: Représente les données d'un profil utilisateur avec ses allergènes.
- **ProduitAlimentaire**: Contient les informations d'un produit alimentaire et ses ingrédients.
- **RésultatAnalyse**: Stocke les résultats d'une analyse d'allergènes pour un produit.

4. Bases de Données et API

Ces composants gèrent le stockage, la persistance et l'accès aux données externes:

- **BD_Firebase**: Base de données cloud pour le stockage persistant des données utilisateurs.
- **BD_CacheLocal**: Stockage local sur l'appareil pour les données fréquemment utilisées.
- **API_OpenFoodFacts**: Interface avec la base de données externe OpenFoodFacts.

5. Services Externes

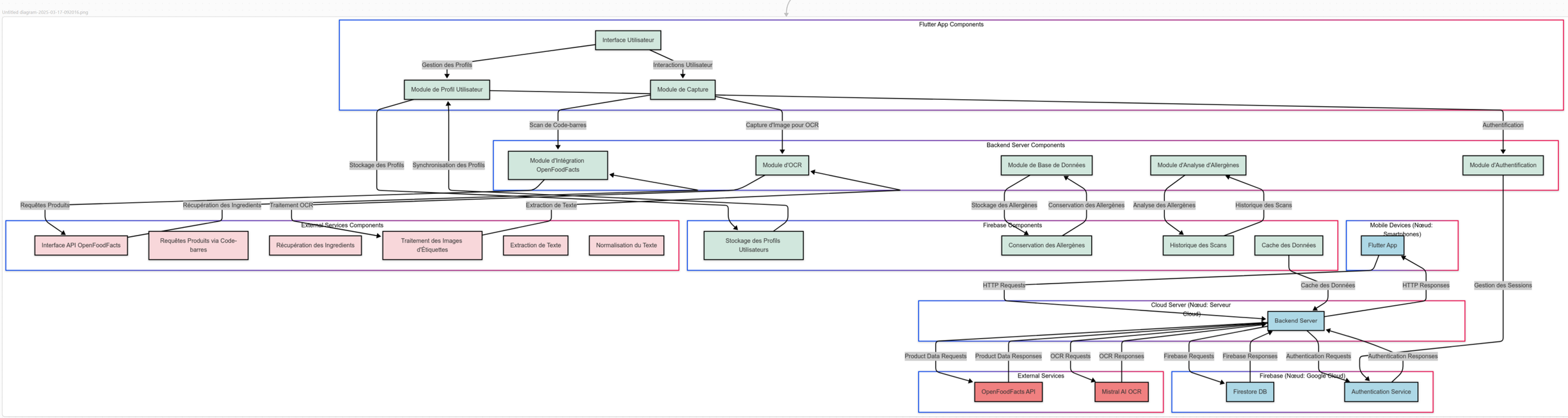
- **ServiceOCR_Mistral**: Service d'OCR pour l'extraction de texte à partir d'images d'étiquettes.

6. Services de Données

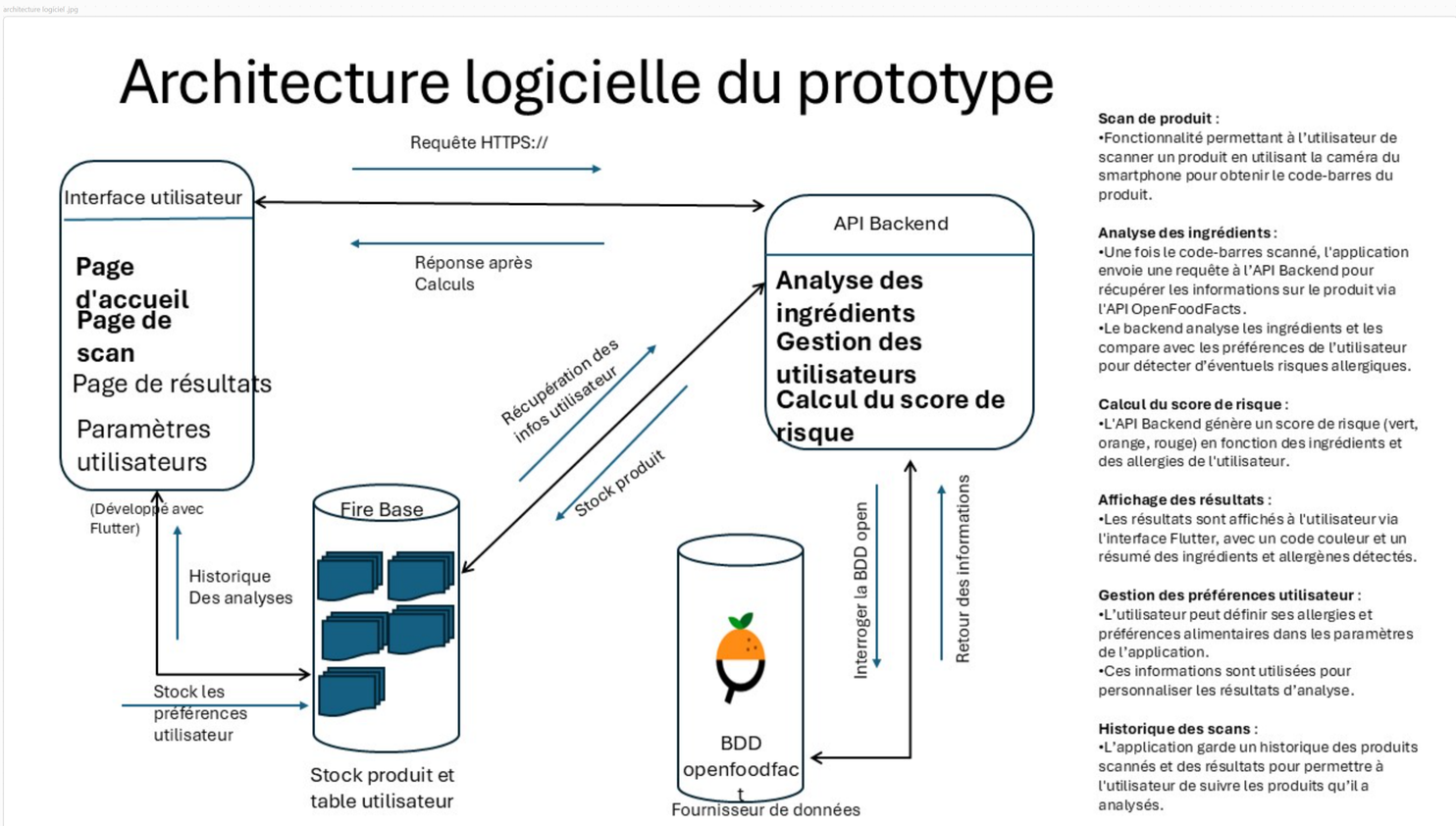
- **ServiceDonnées**: Gère toutes les opérations de persistance et synchronisation des données.

architecture de déploiement de la solution logicielle envisagée

Untitled diagram-2025-03-17-092016.png



architecture logiciel .jpg



Scan de produit :
•Fonctionnalité permettant à l'utilisateur de scanner un produit en utilisant la caméra du smartphone pour obtenir le code-barres du produit.

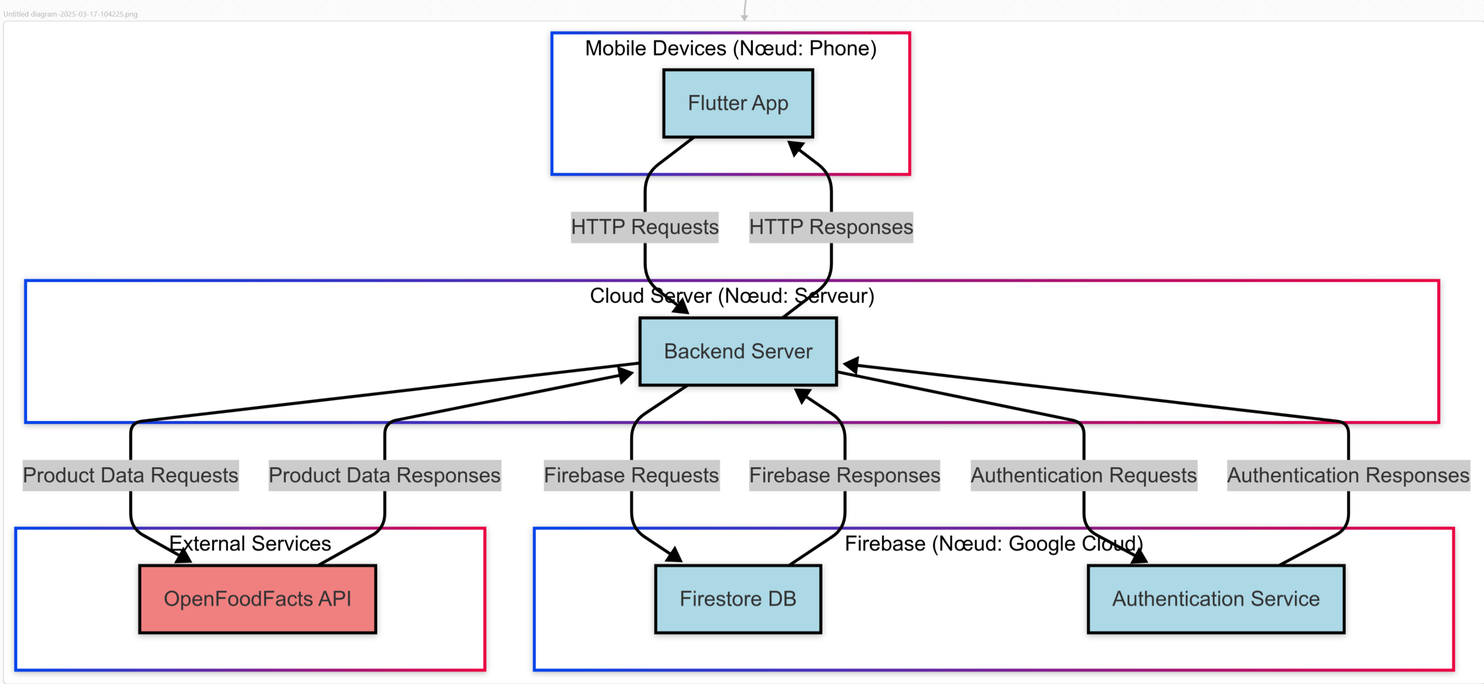
Analyse des ingrédients :
•Une fois le code-barres scanné, l'application envoie une requête à l'API Backend pour récupérer les informations sur le produit via l'API OpenFoodFacts.
•Le backend analyse les ingrédients et les compare avec les préférences de l'utilisateur pour détecter d'éventuels risques allergiques.

Calcul du score de risque :
•L'API Backend génère un score de risque (vert, orange, rouge) en fonction des ingrédients et des allergies de l'utilisateur.

Affichage des résultats :
•Les résultats sont affichés à l'utilisateur via l'interface Flutter, avec un code couleur et un résumé des ingrédients et allergènes détectés.

Gestion des préférences utilisateur :
•L'utilisateur peut définir ses allergies et préférences alimentaires dans les paramètres de l'application.
•Ces informations sont utilisées pour personnaliser les résultats d'analyse.

Historique des scans :
•L'application garde un historique des produits scannés et des résultats pour permettre à l'utilisateur de suivre les produits qu'il a analysés.

architecture de déploiement du prototype

Untitled diagram-2025-03-17-104225.png