# Projet : Alerting monétique (Parent / Enfant)

## 1. Présentation du sujet

Le système d’alerting monétique permet aujourd’hui aux clients de paramétrer des alertes sur leurs comptes bancaires. Les alertes sont gérées au niveau du compte : si plusieurs cartes sont rattachées au même compte, elles héritent toutes du même paramétrage.   
  
Cependant, un problème apparaît lorsque des parents souhaitent paramétrer des alertes pour le compte de leur enfant. Actuellement, cela est impossible car la règle impose que le client dispose d’une carte à son nom sur le compte.

## 2. Problème

Lorsqu’un parent tente de paramétrer une alerte pour le compte de son enfant :  
- Le système vérifie si l’ID client (celui du parent) possède une carte éligible sur le compte de l’enfant.  
- Comme ce n’est pas le cas, le paramétrage échoue avec un message du type : « Non activable faute d’une carte à votre nom sur ce compte ».  
  
Cela empêche les parents de suivre les dépenses de leurs enfants.

## 3. Solution proposée

Introduire un nouvel indicateur « compte\_enfant » qui permettra de différencier les cas classiques des cas parent/enfant.  
  
- Par défaut, l’indicateur est à faux.  
- Lorsqu’un parent paramètre une alerte pour le compte de son enfant, l’application envoie compte\_enfant=true.  
- Le système acceptera alors le paramétrage si le compte de l’enfant dispose d’au moins une carte éligible, même si le parent n’a pas de carte à son nom.

## 4. Tâches à réaliser

- Ajout du champ côté Java (DTO, API, modèle).  
- Développement de la logique de traitement prenant en compte l’indicateur compte\_enfant.  
- Ajout de la colonne « compte\_enfant » dans la base de données (coordination avec un DBA).  
- Génération et adaptation des textes et e-mails (CRM/BPI).  
- Création de comptes parent–enfant pour les tests.

## 5. Questions ouvertes

- La valeur de l’indicateur « compte\_enfant » doit-elle être transmise par BPI en query paramètre ou dans le body ?  
- Comment garantir la bonne identification de la relation parent–enfant (source de vérité) ?  
- Quels sont les cas limites à couvrir (ex. cartes en liste noire, comptes sans carte, multi-cartes) ?

## 6. Macro‑chiffrage (estimation haute/basse)

Estimation indicative en jours-homme (JH) par lot, hors imprévus majeurs. Les fourchettes tiennent compte d’un développement standard, code review et tests fonctionnels de base.

|  |  |  |  |
| --- | --- | --- | --- |
| Lot / Tâche | Effort (JH) | Hypothèses | Dépendances |
| Ajout du champ côté Java (DTO/API/modèle) | 1,5 – 2,5 JH | Un dev back confirmé | — |
| Développement logique d’éligibilité (incl. logs, feature flag) | 2 – 3,5 JH | Règles stables | — |
| Contrats API & exemples (OpenAPI) | 0,5 – 1 JH | Modèle existant | — |
| Ajout colonne DB2 `compte\_enfant` (avec DBA) | 1 – 2 JH | Fenêtre de déploiement OK | DBA |
| Scripts migration/rollback + revue perf/index | 0,5 – 1 JH | Schéma simple | DBA |
| Adaptations BPI/CRM (textes & e‑mails) | 1 – 2 JH | Wording validé rapidement | PO/CRM |
| Jeux de données & comptes parent‑enfant (INT/UAT) | 0,5 – 1 JH | Accès référentiels OK | Ops |
| Tests unitaires & d’intégration | 1,5 – 2,5 JH | Environnements stables | QA/Dev |
| E2E + UAT + corrections mineures | 1,5 – 2,5 JH | Parcours simples | QA/PO |

### Synthèse

Plage totale (hors marge) : \*\*10 – 17 JH\*\*. Ajouter une \*\*marge de management de 20%\*\* pour aléas et synchronisations inter‑équipes.

➡️ Charge macro recommandée : \*\*12 – 20 JH\*\* (~2 à 4 jours ouvrés en focale multi‑ressources, ou 2 à 3 semaines calendrier avec validations DBA/PO/CRM et UAT).

### Hypothèses & risques majeurs

- Relation parent‑enfant déjà disponible et exploitable dans les référentiels.  
- Pas de changement d’UX majeur côté app.  
- Fenêtre DBA et environnements disponibles sans blocage.  
- Risques : retard de validation des textes, indisponibilité des environnements, changement de règles d’éligibilité (liste noire, produits spéciaux).