

Plan de migration

Critères de Décision

Facteurs Techniques

- **Couplages cachés** : Complexité réelle des interdépendances, code et squads
- **Maturité de l'équipe** : Niveau React Query/Zustand et patterns architecturaux
- **Infrastructure CI/CD** : Capacité à gérer la complexité en garantissant la qualité
- **Tests existants** : Qualité et couverture comportementale
- **28 contextes identifiés** → Objectif <10 (réduction 65%) OU selon les usages préconisés (cadre points de 5 identifiés)
- **Hook hell documenté** : useCtaWordingAndAction (complexité 58), useSync.ts
- **Performance critique** : Accueil P95 10-15s → objectif <2s
- **Bundle size** : Android 18.7MB → <15MB, iOS 33.8MB → <25MB

Facteurs Produit

- **Priorités produit** : Performance vs nouvelles features
 - **Pages critiques dégradées** : accueil-thématique (très visitée, LCP 15s)
 - **Résilience insuffisante** : retry: 0, useErrorBoundary: true
 - **UX impactée** : modales nécessitent double-clic, freeze 1s...
- **Tolérance régressions** : Acceptation des risques temporaires
- **Budget tech debt** : Ressources allouées au refactoring
- **Pression délais** : Urgence des livraisons

Facteurs Équipe

- **Niveau technique** : Senior vs mixte
- **Turn-over** : Stabilité de l'équipe
- **Motivation** : Appétence pour le refactoring
- **Formation** : Capacité d'apprentissage nouvelles techno et patterns

Métriques de Succès

Performance (Firebase)

- **App Start Time** : < 2s cold start, < 1s warm start
- **TTI Home** : Offres visibles + scroll possible < 2 s
- **TTI Search** : Input focus + suggestions affichées < 1.5 s
- **TTI Offer** : Image + titre + CTA visibles < 1 s
- **Screen Rendering** : 60 fps maintenu
- **Network Success Rate** : > 95% requests HTTP
- **Memory Usage** : < 150 MB sur devices mid-range Android
- **Bundle size** : Réduction attendue avec suppression contexts, compter 20%
- **Erreurs réseau** : -70% avec retry policy optimisée
- **JS Bundle** : < 5MB (critique pour performance)

Développeur Experience

- **Vélocité équipe** : Temps cycle features
- **Coverage comportementale** : Tests métier vs implémentation
- **Complexité cognitive** : <15 sur fonctions critiques
- **Time to onboard** : Nouveaux développeurs

Impact Produit

- **Taux conversion** : Pages performance améliorées
- **Taux erreur production** : Réduction bugs
- **Satisfaction utilisateur** : UX metrics
- **Time to market** : Nouvelles features

Phases

Phase 1: Fondation et Formation (T3 - 2 mois)

Objectif : Établir nouvelles fondations sans casser l'existant et former les développeurs aux principes

Actions prioritaires gérées par Tech Lead ou Guilde Archi :

- React Query configuration optimisée

```

1  # 1. React Query config fix
2  # src/libs/react-query/queryClient.ts
3  {
4    retry: 3,
5    useErrorBoundary: false,
6    staleTime: 5 * 60 * 1000, // 5min
7  }
8
9  # 2. Bundle analyzer CI
10 npm install --save-dev webpack-bundle-analyzer
11 # Setup GitHub Actions alert si >20MB

```

- pain point résolu : erreurs réseaux -50%
- Bundle analyzer setup pour baseline et performances
 - Avoir une baseline
- ESLint rules anti-prolifération des contextes et autres règles de restriction et d'accompagnement
 - `"no-new-contexts": "error"`
- Finir les templates et la documentation des nouveaux patterns, Page/Container/Components avec exemples Pass Culture
- Faire un POC de refactor d'un context pour illustrer la méthode (CulturalSurveyContextProvider ?)

Pain points adressés :

- Environnement test instable → Setup React Query stable
- Pas de baseline performance → Bundle analyzer + métriques avec la guilde et les autres OKR du trimestre
- Formation de l'équipe

🛠 Les leads et la guilde archi, les développeurs seniors, sont là pour accompagner les squads.

Phase 2: Core Refactor (T4-T1 - 4 mois en J/Homme) - T4 en squad

Objectif : Migrer les contexts les plus utilisés

Migrations prioritaires :

- **AuthWrapper** (80 usages) → React Query (effort L)
 - **Pain point** : Refresh token + erreurs auth
 - **Principe appliqué** : React Query pour état serveur
 - **Métrique** : -50% erreurs refresh_access_token
- **LocationWrapper** (~30 usages) → Zustand (effort M)
 - **Pain point** : Permissions + géolocalisation
 - **Principe appliqué** : Zustand pour état local UI
 - **Métrique** : -30% bugs géolocalisation
- **SettingsWrapper** (~15 usages) → React Query (effort S)
 - **Pain point** : Synchronisation settings
 - **Principe appliqué** : React Query cache + persistance
 - **Métrique** : Consistency settings garantie

Refactors architecturaux

useSync.ts elimination (effort XL)

- **Pain point critique** : Source de bugs majeure


- **Principe appliqué** : URL source de vérité + état Zustand
- **Approche** : Feature-driven par page recherche

useCtaWordingAndAction simplification (effort L)

- **Complexité cognitive** 58 → < 15 objectif
- **Principe appliqué** : Logique backend + fonctions pures
- **Approche** : Tests comportementaux d'abord

Pain points adressés :

- 28 contexts → ~5 à 10 contexts (-50%)
- Performance P95 améliorée (auth + location optimisés)
- Maintenance réduite

 Il serait possible de diviser le travail par squad et d'en faire des objectifs pour fin T4

- SettingsWrapper + AuthWrapper ➡ Activation ? S + L
- LocationWrapper + useCtaWordingAndAction ➡ Conversion ? M + L
- useSync ➡ Découverte ? XL

Check de fin

- ☐ Config React Query déployée + métriques erreur réseau
- ☐ 3 contexts migrés (Auth, Location, Settings)
- ☐ Tests environment stable confirmé
- ☐ Bundle analyzer intégré CI/CD

Phase 3: Feature-Driven (T1-T2 2026 - 6 mois en J/Homme) - T1 en squad

Objectif : Refactor complet par slices verticales (feature driven)


Refactors majeurs :

- Search complet (SearchWrapper + useSync)

- **Pain point** : useSync + SearchWrapper qui sont d'une complexité majeure
- **Approche** : URL source vérité + Zustand + React Query...
- **Métriques** : -50% bugs search + dev velocity +50%, améliorer le P95 de temps d'affichage de la recherche
- Home performance optimization
 - **Pain point** : TTI (P95) proche des 6s
 - **Approche** : Principalement le principe 5 (backend pre-computed) + lazy loading + archi
 - **Métriques** : P95 < 2s + bundle reduction
- Offer display et CTA à simplifier
 - **Pain point** : Complexité cognitive 58 + performance de la modale
 - Approche** : Backend logic + tests comportementaux + archi
 - Métriques** : Suppression double-clic + maintenabilité, améliorer le P95 de temps d'affichage de l'offre

Pain points adressés :

- Contexts count : 15 → <10 (objectif final)
- Performance : P95 <2s atteint
- Bundle size : objectifs Vision 2025 atteints avec 6 mois de « retard » max
- Maintenabilité : Tests robustes + architecture claire

 Il serait possible de diviser le travail par squad et d'en faire des objectifs pour fin T1

- Home performance optimization ➡ Activation
- Search complet (SearchWrapper + useSync) ➡ Conversion
- Offer CTA simplification ➡ Découverte

Check de fin

☐ useSync.ts éliminé + tests search robustes

☐ Performance P95 <2s atteint pages critiques

☐ Hook complexity <15 sur fonctions critiques

☐ Design system Button/Link consolidé