# 🌐 Grille d’analyse du développeur React et des performances applicatives

## 🎯 Mission générale

> Concevoir, développer et maintenir des interfaces utilisateurs efficaces, lisibles et évolutives, en conciliant les impératifs **techniques**, **produits** et **humains**, dans un **écosystème React** et ses extensions (librairies, architectures, outils).

---

## 🧩 Missions par niveau de maturité

| Niveau | Mission principale | Focus |
|---|---|---|
| **Junior** | Traduire des maquettes en composants fonctionnels, comprendre les bases du rendu React et manipuler l’état local et les props de manière simple. | **Productivity**, **Onboarding**, **Execution** |
| **Middle** | Structurer des interfaces complexes, intégrer des données distantes, adopter des patterns de composition, contribuer à la lisibilité du code commun. | **Maintainability**, **Extensibility**, **Evolution** |
| **Senior** | Concevoir l’architecture front-end, standardiser les patterns, anticiper les dettes techniques, encadrer l’équipe et prendre des décisions d’arbitrage en fonction des besoins business et techniques. | **All facets**, avec vision produit et **mentoring** |

---

## 🛠️ Compétences attendues par niveau

| Compétence | Junior | Middle | Senior |
|---|---|---|---|
| **JSX & Components** | Utilisation simple | Composition avancée | Abstraction, composabilité, HOC/renderless |
| **State Management** | `useState`, `useEffect` | `useReducer`, `Context`, libs externes | Choix stratégiques : Redux, Zustand, archi hexagonale |
| **Routing** | Notions | Intégration de React Router | Architecture navigation, lazy routes |
| **Data Fetching** | `fetch`, axios | React Query, gestion du cache | Cohérence des sources de vérité |
| **Performance UI** | Bonne pratique (memo, key) | Optimisation ciblée (`useMemo`, lazy…) | Audit, métriques, split bundles |
| **Testing** | - | Unit tests (Jest, RTL) | Coverage, e2e, TDD |
| **Architecture & Patterns** | - | SoC, container/presentational | Design System, DDD, CQRS, stratégie migration |

| Compétence | Junior | Middle | Senior |
|---|---|---|---|
| **Soft skills** | Apprendre, poser des questions | S'intégrer, documenter, reviewer | Mentor, arbitrer, vulgariser |

## ⚖️ Synthèse des postures

| Niveau | Vision | Responsabilité | Contribution clé |
|---|---|---|---|
| Junior | Locale | Exécution | Rapidité d'implémentation |
| Middle | Transverse | Structure | Mise en cohérence |
| Senior | Systémique | Décisionnelle | Durabilité produit |

## 📐 Les 6 facettes de la performance applicative

| Facette | Définition | React par défaut | Besoin d'extension ? |
|---|---|---|---|
| **Execution** | Temps de réponse, fluidité de l'UI | ✅ Virtual DOM, granularité | `memo`, `lazy`, `useCallback`… |
| **Productivity** | Facilité et rapidité de développement | ✅ Haute au début | Nécessite des patterns dès que complexité |
| **Extensibility** | Ajout de nouvelles fonctionnalités | ✅ Composants + Hooks | Fragile sans cadre |
| **Maintainability** | Lisibilité, robustesse, dette technique | Variable selon rigueur | Architecture, patterns, conventions |
| **Onboarding** | Intégration de nouveaux développeurs | ✅ Rapide sur petits projets | ❌ Difficile sans cohérence |
| **Evolution** | Évolution sans dette, adaptation au changement | React stable | ❌ Pas de support d'architecture métier |

## 🔍 Performance UI : un besoin **contextuel**, non absolu

> La performance UI est importante mais **pas toujours critique** :

- App interne métier → lisibilité & robustesse > fluidité
- App critique (santé, finance) → traçabilité > réactivité
- MVP / prototype → rapidité dev > optimisation
- Grand public (B2C) → UI fluide = avantage concurrentiel

🧠 **Résumé** : toujours évaluer les priorités **contexte-projet** avant d'optimiser prématurément.

## 📌 Grille d'arbitrage (React + Patterns)

| Problème | Facette visée | Exemple de solution externe |
|---|---|---|
| Accès API + cache | Execution, Evolution | React Query, SWR, Apollo |
| Organisation métier | Maintainability | DDD, Repository Pattern |
| Flux d'état complexe | Extensibility | Redux, Zustand, Jotai |
| Navigation | Onboarding, Evolution | React Router, TanStack Router |
| UI complexe | Productivity | Design System, Storybook |