# **Choix Techniques**

# **Introduction**

Thinkshare est une plateforme mobile et web dédiée à la publication de contenu premium par des créateurs, intégrant des fonctionnalités d’abonnement, d’interaction sociale, de gestion de contenu et de paiement sécurisé. Dans ce document, nous présentons les choix technologiques retenus, les alternatives envisagées, ainsi que les avantages techniques des outils imposés pour le projet : **Flutter** (frontend) et **Golang** (backend).

## **Alternatives Envisagées**

Avant de valider Flutter et Go, plusieurs technologies ont été considérées :

### **1.Backend**

* **Node.js (JavaScript/TypeScript)**
  + ✅ Très populaire, riche écosystème de packages (npm), bon pour prototypage rapide
  + ❌ Moins performant pour la concurrence intensive ou les traitements lourds
* **Ruby on Rails**
  + ✅ Productivité rapide, architecture MVC claire
  + ❌ Peu adapté aux applications modernes nécessitant performance/concurrence
* **PHP (Laravel)**
  + ✅ Facile à héberger, bonne documentation
  + ❌ Moins performant et moins évolutif pour une architecture moderne

### **2.Frontend Mobile/Web**

* **React Native**
  + ✅ Large communauté, code JavaScript unifié, bonne intégration avec l’écosystème React
  + ❌ Moins performant sur UI avancée, dépend fortement de ponts natifs
* **Ionic (Angular/React)**
  + ✅ Facile à prendre en main, composants natifs simulés
  + ❌ UI parfois moins fluide, dépendance au WebView

## **Technologies Retenues**

### **Flutter (Frontend Web & Mobile)**

Flutter est un framework développé par Google permettant de créer des applications natives pour **Android**, **iOS**, **Web** et **desktop** à partir d’un seul codebase écrit en **Dart**.

#### **Avantages techniques :**

* **Compilation en code natif (AOT)** : Flutter compile le code Dart en code machine natif, éliminant le besoin d'interprétation à l'exécution, ce qui se traduit par des temps de démarrage plus rapides et une exécution plus fluide.
* **Moteur de rendu Skia** : Flutter utilise son propre moteur de rendu, Skia, pour dessiner l'interface utilisateur directement sur le canevas, offrant des performances graphiques élevées et une cohérence visuelle sur toutes les plateformes.
* **Animations fluides** : Grâce à son architecture optimisée, Flutter peut rendre des animations complexes à 60 FPS, voire 120 FPS sur des appareils compatibles, assurant une expérience utilisateur fluide.
* **Gestion efficace de la mémoire** : Le langage Dart de Flutter gère efficacement l'allocation de mémoire et le garbage collection, garantissant que les applications restent réactives même pendant des tâches intensives.
* **Profiling et outils de performance** : Flutter offre une gamme d'outils de profilage pour mesurer et optimiser les performances des applications, y compris l'utilisation du CPU et de la mémoire, les taux de trame et les requêtes réseau.
* **Multi-plateforme** : Un seul codebase pour Web, Android, iOS → gain de temps et de maintenance
* **UI flexible** : Contrôle total sur les composants d’interface, utile pour design personnalisés
* **Tests et maintenance** : Bon support pour tests d’intégration/UI
* **Système de thèmes** intégré : facilite l’harmonisation visuelle

#### **Cas d’usage pour Thinkshare :**

* Interface utilisateur dynamique pour le fil de contenu
* Espace créateur avec dashboard
* Notifications et interactions temps réel (avec sockets ou Firebase)
* Version web simplifiée pour accès rapide via navigateur

### **Go (Golang – Backend)**

Golang est un langage compilé, open source, développé par Google, conçu pour la performance, la simplicité et la gestion efficace de la concurrence.

#### **Avantages techniques :**

* **Concurrence native (goroutines)** : Go implémente un modèle de concurrence basé sur les goroutines, des threads légers gérés par le runtime de Go, permettant de gérer efficacement des milliers de tâches simultanément avec une surcharge minimale.
* **Modèle de threading M:N** : Go utilise un modèle de threading M:N, où plusieurs goroutines sont multiplexées sur un nombre moindre de threads OS, optimisant l'utilisation des ressources système et améliorant la performance globale.
* **Gestion efficace de la mémoire** : Go dispose d'un ramasse-miettes (garbage collector) optimisé pour minimiser la latence, ce qui est essentiel pour les applications en temps réel.
* **Planificateur de tâches performant** : Le planificateur de Go utilise des techniques telles que le vol de travail (work stealing) et les threads tournants (spinning threads) pour équilibrer la charge entre les processeurs, réduisant ainsi la surcharge et améliorant l'efficacité.
* **Compilation en code natif** : Go compile directement en code machine, offrant des performances d'exécution rapides et une empreinte mémoire réduite.
* **Simplicité du code** : syntaxe claire, architecture modulaire facile à maintenir
* **Sécurité et fiabilité** : Très utilisé dans des systèmes critiques (Docker, Kubernetes)
* **Microservices-ready** : Go est parfaitement adapté à une architecture distribuée avec services indépendants
* **Tests unitaires natifs** : structure de test intégrée dans le langage
* **Logs formatés et contrôlables**

#### **Cas d’usage pour Thinkshare :**

* Gestion des utilisateurs, rôles et authentification (JWT, OAuth2)
* Publication/CRUD de contenu avec droits d’accès
* Traitement des paiements et abonnement (intégration Stripe, webhook)
* Tableau de bord avec génération de statistiques filtrables
* Administration (modération, signalement, logs, monitoring)

## **Couplage Flutter ↔ Go**

L’intégration Flutter avec Go se fait via **API REST** ou **WebSockets** :

* Flutter appelle les endpoints sécurisés exposés par Go (backend RESTful)
* Les deux technologies sont performantes, légères, modernes, bien documentées

### **Stack proposée :**

* Flutter : UI/UX (mobile et web)
* Go : API backend (REST), gestion métier

## **Conclusion**

Les choix de Flutter et Go pour le projet Thinkshare s’inscrivent dans une logique de performance, de maintenabilité, et de développement fullstack moderne. Ce duo permet de construire une application mobile et web avec une interface fluide et une architecture backend robuste, scalable et sécurisée.

Ces choix nous donnent également une maîtrise complète du cycle de développement, en respectant les contraintes de tests, de modularité, et de mise en production attendues dans un environnement professionnel.