ChatGPT

Prompt pour générer l'application Flutter (Compatibilité & Guidance amoureuse)

Ce document sert de **brief technique** destiné à un modèle générateur de code (Codex) pour créer un projet Flutter complet basé sur les documents `inputs_logic.md`, `interpretations.md` et `ui_flow.md`. L'objectif est de produire une application mobile simple, esthétique et modulaire permettant de calculer la compatibilité de couple, de donner une guidance quotidienne et d'inciter les utilisateurs à revenir.

1. Structures et bibliothèques

- **Framework :** Flutter (SDK stable au moment de la génération). Utiliser les conventions de projet standard (`lib/main.dart`, `pubspec.yaml`, etc.).
- **Packages recommandés :**
- `flutter_hooks` ou `provider` pour la gestion de l'état.
- `stepper` ou `step_progress_indicator` pour afficher une barre de progression ou un Stepper, permettant de découper le formulaire [1].
- `shared_preferences` pour enregistrer localement les données et permettre une reprise sans backend.
- `intl` pour manipuler les dates et formater les jours, mois et années.
- Facultatif : `supabase_flutter` si un backend est requis plus tard (en laissant des méthodes abstraites pour l'envoi des données).

2. Architecture de l'application

- **Écrans (Pages)** :
- **WelcomePage** : écran d'accueil avec message d'introduction, illustration et bouton « Commencer ».
- **NamesPage** : collecte des prénoms et du genre (facultatif). Comprend des `TextFormField` avec validation.
- **BirthdatesPage** : sélection des dates de naissance via des `DatePicker` séparés pour chaque partenaire.
- **ContextPage** : questions facultatives (date de rencontre, durée, défis principaux) implémentées via des `DropdownButton`, `CheckboxListTile` et `DatePicker`.
- **ContactPage** : champ e-mail et case à cocher pour s'abonner aux notifications. Validation d'adresse e-mail.
- **ResultsPage** : affichage des résultats calculés (voir section 4) et CTA pour consulter la guidance quotidienne.

- **DailyGuidancePage** (ou modal) : montre le conseil du jour, calculé à partir de la date actuelle et des nombres personnels, et propose de consulter la prévision du lendemain.

- **Navigation** : utiliser `Navigator` ou un package comme `go_router` pour la navigation. Chaque étape utilise un `Stepper` vertical ou horizontal, synchronisé avec la barre de progression [1]. Prévoir des contrôles « Suivant » et « Précédent ».

- **Stockage local** : encapsuler la logique d'enregistrement et de récupération dans un service `LocalStorageService` utilisant `SharedPreferences`. Sauvegarder :

• prénoms et dates de naissance
• réponses aux questions facultatives
• adresse e-mail et préférences de notification

• date de dernière consultation pour le conseil du jour

- **Backend (facultatif)** : prévoir un `ApiService` et un `DatabaseService` abstraits pouvant être remplacés par Supabase. Ce service devra envoyer les données (prénoms, dates, e-mail) pour stockage ou génération de rapports PDF.

3. Logique métier

La logique de calcul doit reproduire fidèlement les formules décrites dans `inputs_logic.md` :

- **Table de conversion des lettres en nombres :** stocker dans une constante `Map<String, int>` la correspondance de A = 1, B = 2, …, I = 9, J = 1, …, Z = 8.
- **Réduction :** créer une fonction `int reduceToDigit(int n)` qui additionne les chiffres d'un nombre jusqu'à obtenir un chiffre compris entre 1 et 9 (sauf 11, 22, 33). Gérer séparément les maîtres nombres.
- **Calcul du nombre de couple :** additionner les valeurs des lettres des deux prénoms et réduire selon la méthode ci-dessus.
- **Chemin de vie :** convertir la date de naissance en un seul chiffre selon la méthode décrite : additionner jour, mois et année et réduire, en conservant 11, 22 ou 33.
- **Année personnelle, mois personnel et jour personnel :** implémenter les formules détaillées dans `inputs_logic.md` pour calculer ces cycles. Le jour personnel devrait se mettre à jour en fonction de la date actuelle (utiliser `DateTime.now()` et le fuseau horaire de l'utilisateur). L'algorithme repose sur l'addition de la date de naissance et de la date en cours, puis réduction [2].
- **Nombre kabbalistique :** fonction qui calcule le reste de la division par 9 du total des valeurs de lettres d'un nom (alphabet latin 1–26), comme indiqué dans le document [3].
- **Interprétations :** charger les interprétations depuis un fichier JSON ou une classe Dart (`Map`) basé sur `interpretations.md`. Associer chaque chiffre (couple, année, chemin de vie, nombre kabbalistique) à son texte correspondant.

4. Présentation des résultats

Sur la page de résultats :

1. **Compatibilité de couple** : afficher le numéro obtenu et sa description (ex. couple 1 = passionnel avec risque d'ennui). Mettre en avant les points forts et les points de vigilance.
2. **Profils individuels** : afficher pour chaque partenaire son chemin de vie (numéro et interprétation), son nombre kabbalistique et, éventuellement, sa vibration d'expression/aspiration si implémentée.
3. **Prévisions de l'année personnelle** : expliquer brièvement l'influence de l'année en cours (années 1 à 9) sur la relation (travail, amour, argent, famille, santé) en utilisant les interprétations fournies.

4. **Conseil du jour** : calculer le nombre du jour pour le couple (ou séparément) et afficher un court message incitant ou déconseillant certaines actions. Mettre en évidence que les cycles changent régulièrement afin d'encourager l'utilisateur à revenir [4] .
5. **Appel à l'action** : proposer un bouton « Voir ma vibration de demain » ou « Recevoir mes prévisions chaque jour » menant à un système d'abonnement ou à l'enregistrement local d'un rappel.

5. Design et UI

- Utiliser un thème clair et attrayant avec des couleurs douces (pastels) et des icônes inspirées de la numérologie et de la spiritualité (cœurs, étoiles, chiffres).
- Chaque page doit être accessible et mobile-friendly. S'assurer que les champs de formulaire sont suffisamment espacés et utilisables sur mobile.
- La barre de progression/Stepper doit être placée en haut de l'écran et mise à jour à chaque étape. Utiliser des couleurs ou des icônes pour indiquer les étapes complétées [5] .
- Tous les champs obligatoires doivent être clairement marqués et accompagnés d'un label. Les placeholders ne doivent pas remplacer les labels [6] .
- Prévoir des animations douces lors des transitions entre les pages pour rendre l'expérience agréable.

6. Fonctionnalités optionnelles

- **Supabase** : si un backend est activé, créer des fonctions asynchrones pour envoyer les données des utilisateurs (prénoms, dates, e-mail) à une table Supabase et récupérer une analyse générée côté serveur. Prévoir un système d'authentification par e-mail pour permettre à l'utilisateur de sauvegarder et consulter ses analyses sur plusieurs appareils.
- **PDF/rapport** : générer un PDF récapitulatif à partir des résultats et l'envoyer par e-mail ou le rendre téléchargeable. Ce module peut s'appuyer sur la bibliothèque `pdf` de Flutter.
- **Notifs push** : intégrer `firebase_messaging` pour envoyer une notification quotidienne avec le conseil du jour.

7. Instructions finales

- Le code doit être proprement commenté en français, en expliquant chaque fonction et étape clé.
- Organiser les fichiers dans des dossiers (ex. `lib/screens`, `lib/models`, `lib/services`, `lib/widgets`) pour une meilleure maintenabilité.
- Prévoir des tests unitaires pour la logique de calcul (réduction, couple number, chemin de vie, etc.).
- Documenter dans un fichier `README.md` la procédure d'installation, de compilation et de lancement (y compris la configuration éventuelle de Supabase).

En suivant ces spécifications, le modèle Codex devrait générer une application Flutter complète et fonctionnelle, alignée sur les besoins du projet et les meilleures pratiques UX identifiées [4] [5] .

---

[1] Creating a multi-step form in Flutter using the Stepper widget - LogRocket Blog
https://blog.logrocket.com/creating-multi-step-form-flutter-stepper-widget/

[2] Your Personal Years, Months, and Days
https://affinitynumerology.com/using-numerology/your-personal-years-months-and-days.php

3 Kabbalah Numerology
https://numerology.center/kabbalah_numerology.php

4 6 Must-Follow UX Best Practices When Designing A Multi Step Form - Growform Multi Step Form Builder
https://www.growform.co/must-follow-ux-best-practices-when-designing-a-multi-step-form/

5 Design Better Progress Trackers and Control User Expectations
https://www.uxpin.com/studio/blog/design-progress-trackers/