Projet Flutter 02— Plateforme de gestion de projets collaboratifs

Nom du projet : SUP4 DEV – FlutterTrello

Objectif :

Créer une application mobile Flutter **type Trello simplifié**, où les utilisateurs peuvent :

- gérer leurs projets,
- ajouter et suivre les tâches dans un tableau (kanban) ou une liste,
- collaborer avec d'autres utilisateurs.

Fonctionnalités attendues

Utilisateurs

- Authentification (login/signup).
- Rôles : utilisateur, administrateur.
- Gestion des permissions : propriétaire, membre invité, etc.

📁 **Projets**

- Création de projet.
- Invitation de membres (emails fictifs ou UID).
- Définition du chef de projet.

📋 **Tâches**

- Organisation par tableau ou liste.
- Colonnes de type : À faire, En cours, Terminé.
- Attribution de tâches à des utilisateurs.
- Commentaires.
- Pièces jointes (via image_picker, file_picker...).

🕓 **Historique & Activité**

- Fil d'activité du projet.
- Notifications sur les actions importantes.

⚙️ Aspects techniques Flutter

Architecture & Structure

- **Architecture Clean** ou MVC.
- Séparation claire : models/, services/, providers/, views/, widgets/.

Backend

- API REST simulée ou connectée à un backend NodeJS / Symfony / Firebase.
- Possibilité d'utiliser **Firebase Firestore** pour temps réel (comme Trello).

Gestion d'état

- **Provider**, **Riverpod** ou **Bloc** pour la gestion des états des projets et des tâches.

UI/UX

- Interface inspirée de **Trello ou Notion** :
    - Glisser/déposer via flutter_drag_and_drop, flutter_reorderable_list, ou flutter_board.
    - Responsive et fluide.
    - Vue par tableau ou par liste.

Notifications

- Intégration optionnelle de **Firebase Cloud Messaging** pour les notifications.
- Badge ou système de notification interne au projet.

Maquette visuelle de l'application (obligatoire)

Vous devez proposer une maquette visuelle de votre application (réalisée avec Figma, Adobe XD, Balsamiq).

- Fournissez un lien vers la maquette ou intégrez une capture d'écran claire de l'interface prévue.
- La maquette doit refléter la structure de votre application (navigation, organisation des pages, composants essentiels).
- Elle servira de base de réflexion et de développement tout au long du projet.

Cette étape est obligatoire et sera évaluée. Un bonus pourra être accordé si l'implémentation finale est fidèle à la maquette.

Exemple de structure du projet

```
lib/
├── models/          # Projet, Utilisateur, Tâche, Commentaire
├── services/        # AuthService, ProjectService, NotificationService
├── providers/       # projectProvider, authProvider, taskProvider
├── views/
│   ├── auth/        # Login, Signup
│   ├── dashboard/   # Liste des projets
│   ├── project/     # Détail projet, tableau Kanban
│   └── admin/       # Interface admin
├── widgets/         # CardTache, ColumnBoard, ModalCommentaire
└── main.dart
```

Critères pour valider et recommander le projet sur GitHub

Critère	**Attendu pour validation**
Durée	1 mois de développement
Commits	20 commits minimum
Fonctionnalités	Auth, projet, tâches, permissions, activité
Architecture	MVC / Clean Architecture bien respectée
UI/UX	Interface Kanban/Liste propre et fluide
Code	Lisible, modulaire, bien structuré
Déploiement	Démo APK ou hébergement web (Firebase Hosting, etc.)
Tests (bonus)	Quelques tests unitaires ou widget tests
Sécurité (bonus)	Auth Firebase ou sécurisation REST API

Bonus possibles pour briller

- Glisser/déposer dynamique (drag-and-drop).
- Recherche de tâches.
- Statistiques de productivité (nb de tâches / statut).
- Intégration d’un calendrier.
- Personnalisation des tableaux (couleurs, fonds, etc.).