**Documentation du Projet : Application de Gestion des Ressources (Employés et Stocks) d'une entreprise**

**Introduction**

Cette documentation présente une application mobile de gestion des ressources développée avec Flutter, conçue pour aider les entreprises à optimiser la gestion de leurs employés et de leurs stocks. Dans un contexte économique où l'efficacité opérationnelle est primordiale, cette solution offre une interface intuitive et performante permettant aux gestionnaires de suivre en temps réel l'ensemble des ressources de l'entreprise.

L'application répond aux besoins essentiels des entreprises modernes en centralisant les informations relatives aux employés (nom, rôle, horaires) et aux stocks (quantité, catégorie, prix) au sein d'une seule plateforme facile d'utilisation. Grâce à l'utilisation de technologies avancées comme Flutter et Firebase, cette solution est à la fois réactive, fiable et accessible sur différentes plateformes (iOS et Android).

**Description du Projet**

L'application de gestion des ressources est une solution Flutter permettant de gérer les employés et les stocks d'une entreprise. Elle inclut des fonctionnalités telles que :

* L'affichage des employés et des stocks
* La gestion des employés (ajout, modification, suppression)
* La gestion des stocks (ajout, modification, suppression)

**Technologies Utilisées**

* **Flutter** : Framework pour le développement d'applications multiplateformes
* **Firebase Firestore** : Base de données NoSQL pour stocker les informations des employés et des stocks
* **GetX** : Gestionnaire d'état et de navigation
* **Dart** : Langage de programmation utilisé avec Flutter

**Structure du Projet**

Voici les principaux fichiers et leur rôle dans le projet :

**1. tableau\_de\_bord.dart**

* Affiche le tableau de bord principal avec :
  + Le nombre total d'employés
  + Le nombre total de stocks

**2. home\_page.dart**

* Affiche la liste des employés sous forme de grille
* Permet de naviguer vers le profil d'un employé

**3. home\_page\_stock.dart**

* Affiche la liste des stocks sous forme de grille
* Permet de naviguer vers les détails d'un stock

**4. formulaire\_ajout\_d\_employe.dart**

* Permet d'ajouter un nouvel employé à la base de données Firestore

**5. formulaire\_ajout\_de\_stock.dart**

* Permet d'ajouter un nouveau stock à la base de données Firestore

**6. profile.dart**

* Affiche les informations détaillées d'un utilisateur

**Étapes de Développement**

**1. Configuration de Firebase**

* Créez un projet Firebase
* Ajoutez l'application Flutter à Firebase
* Configurez les fichiers google-services.json (Android) et GoogleService-Info.plist (iOS)
* Activez Firestore dans la console Firebase

**2. Création des Modèles**

**Modèle Employé (Employe)** :

class Employe {

final String nom;

final String prenom;

final String role;

final String heure;

final String image;

final bool travail;

Employe({

required this.nom,

required this.prenom,

required this.role,

required this.heure,

required this.image,

required this.travail,

});

factory Employe.fromMap(Map<String, dynamic> data) {

return Employe(

nom: data['nom'],

prenom: data['prenom'],

role: data['role'],

heure: data['heure'],

image: data['image'],

travail: data['travail'],

);

}

Map<String, dynamic> toMap() {

return {

'nom': nom,

'prenom': prenom,

'role': role,

'heure': heure,

'image': image,

'travail': travail,

};

}

}

**Modèle Stock (Stock)** :

class Stock {

final String nom;

final String categorie;

final int quantite;

final double prixunitair;

final String image;

Stock({

required this.nom,

required this.categorie,

required this.quantite,

required this.prixunitair,

required this.image,

});

factory Stock.fromMap(Map<String, dynamic> data) {

return Stock(

nom: data['nom'],

categorie: data['categorie'],

quantite: data['quantite'],

prixunitair: data['prixunitair'],

image: data['image'],

);

}

Map<String, dynamic> toMap() {

return {

'nom': nom,

'categorie': categorie,

'quantite': quantite,

'prixunitair': prixunitair,

'image': image,

};

}

}

**3. Création des Interfaces**

* **Tableau de Bord** :
  + Utilisation de StreamBuilder pour afficher les données en temps réel
  + Affichage des employés qui travaillent aujourd'hui et ceux en congé avec des requêtes Firestore (where)
* **Liste des Employés** :
  + Utilisation de GridView.builder pour afficher les employés sous forme de grille
  + Navigation vers le profil d'un employé avec Navigator.push
* **Formulaires** :
  + Utilisation de TextFormField pour saisir les informations des employés et des stocks
  + Validation des champs avec des méthodes validator

**4. Navigation**

* Utilisation de GetX pour une navigation simple et efficace :

Get.to(() => const HomePage());

**5. Gestion des Données**

* **Ajout** :
  + Ajout d'un employé ou d'un stock dans Firestore avec la méthode set :

FirebaseFirestore.instance.collection("Employes").doc(nom).set(employe.toMap());

* **Modification** :
  + Mise à jour des données avec la méthode update :

FirebaseFirestore.instance.collection("Employes").doc(employe.nom).update(employe.toMap());

* **Suppression** :
  + Suppression d'un document avec la méthode delete :

FirebaseFirestore.instance.collection("Employes").doc(employe.nom).delete();

**Fonctionnalités Clés**

**1. Tableau de Bord**

* Affiche le nombre total d'employés et de stocks
* Liste les employés qui travaillent aujourd'hui et ceux en congé

**2. Gestion des Employés**

* Ajouter un employé
* Modifier les informations d'un employé
* Supprimer un employé

**3. Gestion des Stocks**

* Ajouter un stock
* Modifier les informations d'un stock
* Supprimer un stock

**Instructions pour Lancer le Projet**

1. Clonez le dépôt :

git clone https://github.com/votre-repo.git

cd votre-repo

1. Installez les dépendances :

flutter pub get

1. Lancez l'application :

flutter run

**Améliorations Futures**

* Ajouter une authentification pour sécuriser l'accès
* L'affichage des employés qui travaillent aujourd'hui et ceux en congé
* Implémenter des notifications pour les employés en congé ou les stocks faibles
* Ajouter des graphiques pour visualiser les données

**Conclusion**

Cette application de gestion des ressources représente une solution complète et évolutive pour les entreprises cherchant à optimiser leur gestion interne. Grâce à son architecture modulaire basée sur Flutter et Firebase, elle offre une expérience utilisateur fluide tout en garantissant la sécurité et l'intégrité des données.

Le développement de cette application s'est concentré sur la simplicité d'utilisation et la performance, permettant aux gestionnaires de gagner un temps précieux dans leurs tâches quotidiennes. Les fonctionnalités de suivi des employés et de gestion des stocks sont particulièrement utiles pour améliorer la productivité et réduire les coûts opérationnels.

Les perspectives d'évolution de l'application sont nombreuses, avec notamment l'ajout d'un système d'authentification, d'outils de visualisation de données et de notifications intelligentes. Ces améliorations permettront d'enrichir encore davantage l'expérience utilisateur et de répondre aux besoins croissants des entreprises en matière de digitalisation de leurs processus.

Cette documentation fournit toutes les informations nécessaires pour comprendre, installer et utiliser l'application, ainsi que pour contribuer à son développement futur. L'utilisation de standards et de bonnes pratiques de développement garantit une maintenance facilitée et une évolution harmonieuse du projet.