

# ROUTINE APP

HEURTAULT AUBIN

BACHA AMINE

HERVÉ EWEN



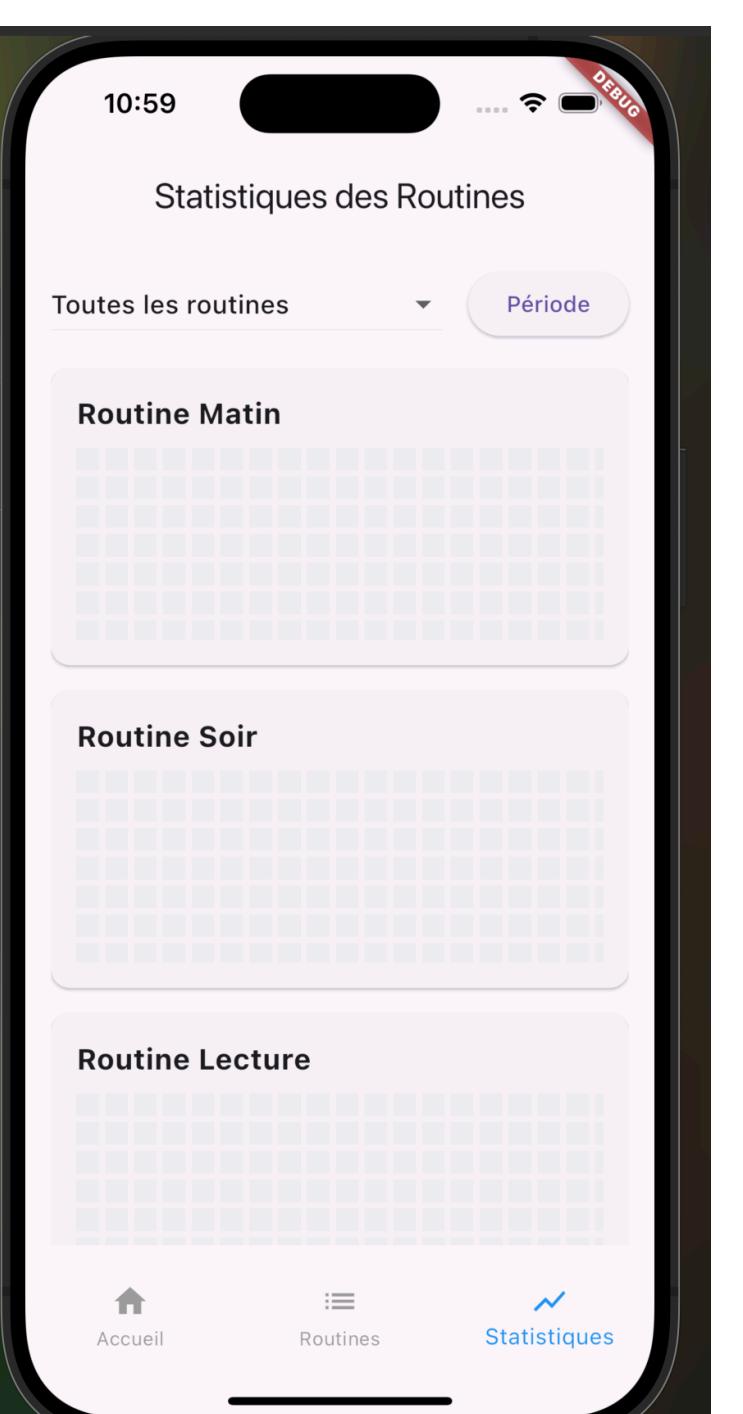
# OBJECTIF DU PROJET

**Créer une application multiplateforme (Android/iOS) pour gérer les routines.**

**Offrir une expérience utilisateur intuitive et un design moderne.**

**Implémenter un système de suivi visuel des routines (inspiré de GitHub).**

**Intégrer un système de notifications pour rappeler les tâches importantes**



# ORGANISATION ET QUALITÉ DU CODE

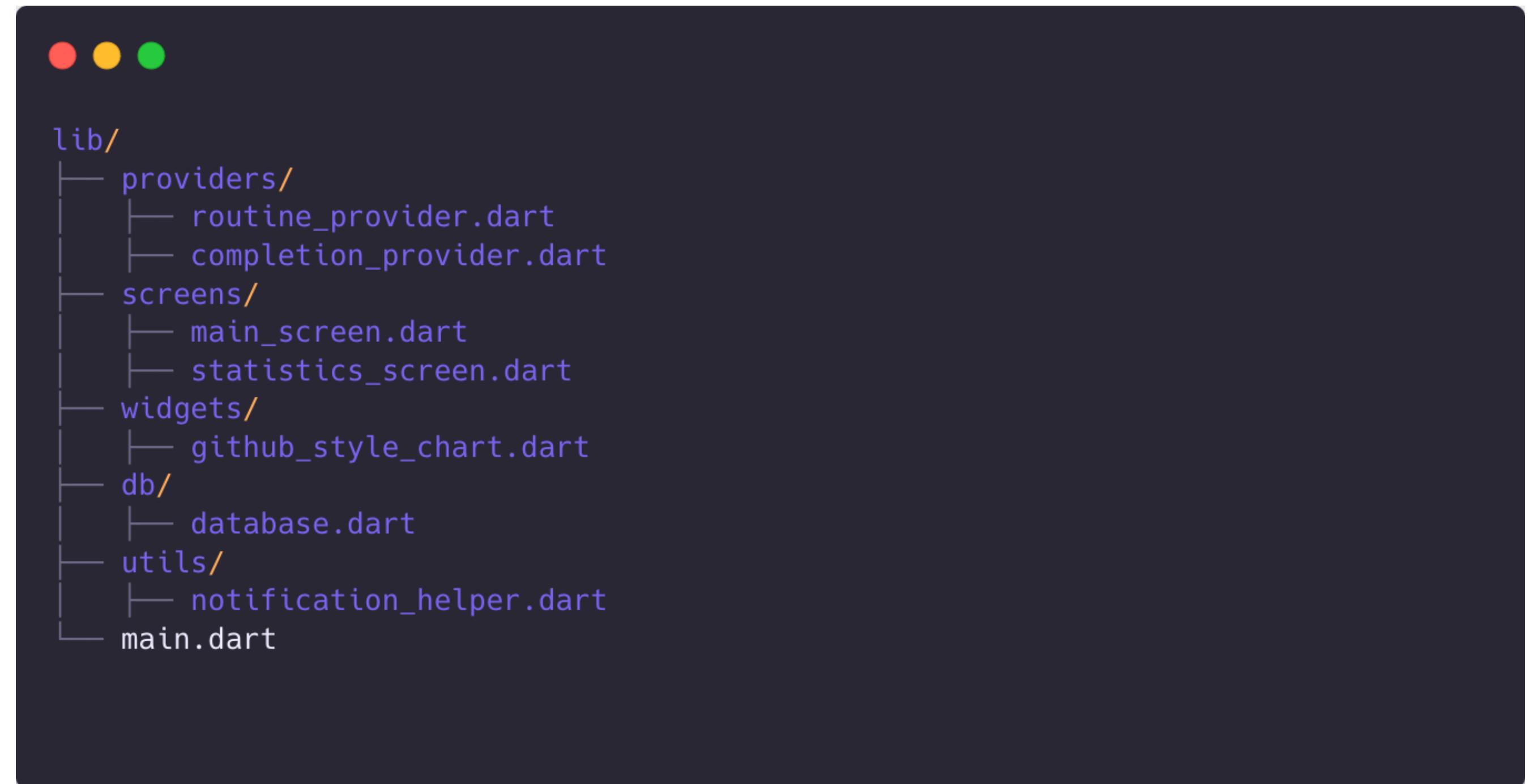
## CHOIX DE L'ARCHITECTURE

### Type-Based :

- Sépare clairement les responsabilités. Par exemple, tous les `Provider` sont regroupés dans un dossier, ce qui facilite leur gestion.
- Idéal pour les projets de petite à moyenne taille où chaque type de fichier joue un rôle distinct.

### Feature-Based :

- Tous les fichiers liés à une fonctionnalité (UI, logic, state) sont regroupés dans un même dossier.
- Peut devenir complexe pour des projets ayant beaucoup de fonctionnalités similaires (par exemple, plusieurs routines ou graphiques).



```
lib/
  providers/
    routine_provider.dart
    completion_provider.dart
  screens/
    main_screen.dart
    statistics_screen.dart
  widgets/
    github_style_chart.dart
  db/
    database.dart
  utils/
    notification_helper.dart
  main.dart
```

# GESTION DES PERMISSION

## NOTIFICATIONS

Permissions nécessaires :

- Notifications pour les rappels.

Gestion des permissions robustes avec :

- `flutter_local_notifications` pour les notifications locales.
- Vérification des autorisations au démarrage de l'app.



```
Future<void> requestNotificationPermission() async {  
    final IOSFlutterLocalNotificationsPlugin? iosPlugin = ...  
    await iosPlugin?.requestPermissions(alert: true, badge: true, sound: true);  
}
```

# PROBLÈMES RENCONTRER

Problème : Lag dans le graphique de progression.

- Solution : Passage à un CustomPainter pour des rendus plus fluides.

Problème calendrier : Crash de l'application a cause d'un plugins

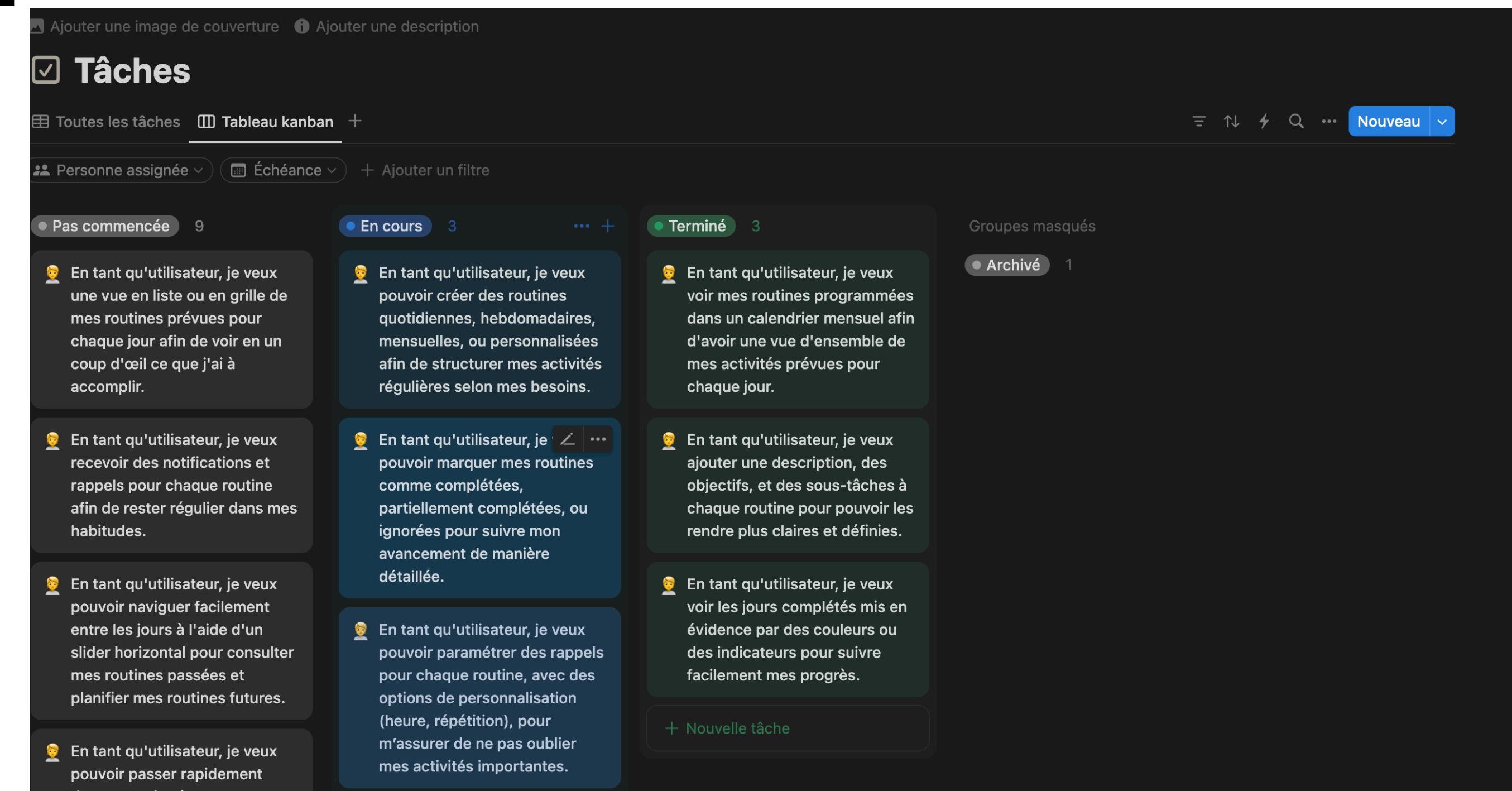
- Solution : Revoir l'affichage et enlever le calendrier

# COLLABORATION

## GESTION DE PROJET

### Organisation des Tâches :

- **Tableau Kanban (To Do, En Cours, Terminé).**
- **Une carte par tâche avec : titre, description, échéance, responsable.**



# COLLABORATION

## GESTION DU CODE

**Création d'une Branche :** Une branche par fonctionnalité ou bugfix, nommée clairement (ex. `feature/graphique`).

**Workflow des PR :**

1. Développer sur la branche.
2. Créer une PR avec description (résumé, impact, tests).
3. Relecture par un collègue (vérification des conventions et bugs).

**Fusion :** Après validation, merge dans `main` ou `develop`, puis suppression de la branche.

The screenshot shows a GitHub interface for the repository 'Eweeen / routine-app'. The 'Pull requests' tab is selected, showing three open pull requests:

- #5: logo (opened 46 minutes ago by Sixpacks123)
- #4: Notifications (opened 1 hour ago by Sixpacks123)
- #3: ajout crud routine + style + validation routine (opened 1 hour ago by aminebacha35)

At the top of the page, there is a notification about labeling issues and pull requests for new contributors. The search bar contains the query `is:pr is:open`. There are also filters for Labels (9), Milestones (0), and a 'New pull request' button.

# **DÉMO ET PRÉSENTATION DU CODE**