



République tunisienne

Ministère de l'enseignement supérieur et de la recherche
scientifique

Université Tunis El Manar

Faculté des sciences de Tunis

Département des sciences de l'informatique

Rapport de projet

Collabflow : Plateforme de gestion de projets

Réalisé par :

Rayen Riahi – Wajdi Ben Abdeljelil

December 14, 2025

Contents

1	Introduction	3
2	Installation du projet	3
2.1	Prérequis	3
2.2	Installation et démarrage	3
3	Architecture / Arborescence	4
3.1	Routing et Authentification	4
3.2	Services Métier	4
3.3	Modèles	5
3.4	Organisation de l'UI	5
3.5	Autres Répertoires	5
3.6	Arborescence	5
4	Description des fonctionnalités	5
4.1	Authentification	5
4.2	Tableau de Bord Admin	6
4.3	Gestion des utilisateurs	6
4.4	Gestion des projets	6
4.5	Planification des réunions	6
4.6	Notifications	6
4.7	Suivi des Développeurs	6
5	Choix techniques	7
5.1	Composants Standalone Angular	7
5.2	Reactive Forms	7
5.3	RxJS	7
5.4	Sécurité Front	7
5.5	Drag & Drop	7
5.6	DOM sécurisé	7
6	Captures d'écran	8
7	Conclusion et limites	12
7.1	Pistes d'amélioration	12

List of Figures

1	Page de connexion avec animation des particules	8
2	Tableau de bord Admin avec les cartes d'indicateurs	8
3	Vue de la gestion des utilisateurs avec filtres et modals	9
4	Vue de la gestion des projets (ajout d'un projet)	9
5	Calendrier des reunions avec drag and drop	10
6	Diagramme de Gantt avec fonctionnalité de drag & drop	10
7	Vue des feedbacks	11
8	Vue des notifications	11

1 Introduction

Ce projet s'inscrit dans le cadre du cours de *Technologies Web et Multimédia* et vise à développer une application de gestion de projets. L'idée générale est de créer une plateforme intuitive permettant de gérer des projets collaboratifs, avec trois types de profils : Administrateur, Manager et Développeur. Chaque profil dispose de fonctionnalités adaptées à ses besoins spécifiques, telles que la gestion des utilisateurs, des tâches et des réunions.

L'objectif de ce projet est de mettre en pratique les concepts étudiés en cours, notamment la création d'interfaces interactives, la gestion des rôles et des autorisations, ainsi que l'intégration de fonctionnalités en temps réel comme les notifications. L'application a été conçue pour démontrer l'application des bonnes pratiques dans le développement front-end, tout en permettant une gestion fluide et collaborative des projets au sein d'une équipe.

2 Installation du projet

2.1 Prérequis

- Node.js version 18 ou supérieure.
- npm (gestionnaire de paquets Node).
- Angular CLI version 19 ou supérieure. Vous pouvez l'installer globalement via la commande suivante :

```
npm i -g @angular/cli
```

2.2 Installation et démarrage

1. Cloner ou télécharger le projet.
2. Installer les dépendances avec npm :

```
npm install
```

3. Démarrer le serveur de développement local :

```
npm start
```

Ensuite, ouvrez votre navigateur et accédez à `http://localhost:4200`.

4. Pour effectuer un build de production :

```
npm run build
```

Le code compilé sera disponible dans le dossier `dist/`.

5. Exécuter les tests unitaires Angular :

```
npm test
```

3 Architecture / Arborescence

L'architecture du projet suit une organisation simple et modulaire, facilitant l'extensibilité et la maintenance. Voici un aperçu des principaux répertoires et fichiers :

3.1 Routing et Authentification

- `src/app/app.routes.ts` : Contient la configuration des routes avec protection par `AuthGuard` et gestion du JWT.

3.2 Services Métier

Les services métiers sont responsables des opérations spécifiques à chaque domaine de gestion de projet. Voici quelques exemples :

- `project.service.ts`
- `task.service.ts`
- `meeting.service.ts`
- `notification.service.ts`

Tous ces services retournent des `Observable` et sont conçus pour offrir une structure flexible et réactive à l'application.

3.3 Modèles

Les modèles utilisés dans l'application sont définis dans `src/app/models/*.ts`, garantissant un typage fort pour chaque objet (par exemple, `Project`, `Task`, `User`, `Meeting`, etc.).

3.4 Organisation de l'UI

- `src/app/admin/` : Gère le login, le tableau de bord, la gestion des utilisateurs et des projets et les rapports.
- `src/app/manager/` : Gère les projets, la planification des réunions, les risques et les feedbacks.
- `src/app/developpeur/` : Gère les tâches et le calendrier des réunions.

3.5 Autres Répertoires

- `assets/` : Contient les ressources statiques telles que les images et les icônes.
- `src/styles.css` : Contient les styles globaux de l'application.

3.6 Arborescence

```
src/  
  app/  
    admin/ | manager/ | developpeur/  
    services/ | models/ | guards/ | interceptors/  
    app.routes.ts | app.component.*  
  assets/
```

4 Description des fonctionnalités

4.1 Authentification

Le formulaire de **connexion** (`LoginComponent`) est réactif et permet une validation côté client du **mail** et du **mot de passe**. Le jeton JWT simulé est stocké localement, et les utilisateurs sont redirigés en fonction de leur rôle après la connexion. Un décor visuel animé avec des particules est ajouté pour une expérience utilisateur plus dynamique.

4.2 Tableau de Bord Admin

Le **tableau de bord Admin** (`DashboardadminComponent`) agrège les données de projets, tâches et utilisateurs à l'aide de l'opérateur `forkJoin`. Ce tableau de bord affiche des indicateurs tels que la répartition des rôles, l'avancement des tâches et la proximité des échéances.

4.3 Gestion des utilisateurs

Le **composant de gestion des utilisateurs** (`GestionUtilisateursComponent`) permet de rechercher, ajouter, éditer et supprimer des utilisateurs. Il inclut une validation des formulaires avec `ReactiveFormsModule` et offre des modals pour une meilleure interaction utilisateur.

4.4 Gestion des projets

Les **projets** sont gérés via `ProjetsComponent`, qui permet de filtrer par statut et date, de rechercher, de modifier les détails des projets et d'ajouter des tâches.

4.5 Planification des réunions

Le **composant de réunions** (`MeetingsComponent`) génère un calendrier interactif avec des fonctionnalités de **drag & drop** grâce à Angular CDK, permettant de planifier, éditer et déplacer des réunions facilement.

4.6 Notifications

Les **notifications** sont gérés par `notification.service.ts`, offrant des fonctionnalités de communication en temps réel via un flux simulé par WebSocket et REST.

4.7 Suivi des Développeurs

Les **tâches** et le **calendrier des réunions** du développeur sont présentés de manière dynamique avec des filtres de statut, des indicateurs de progression, et des notifications en temps réel.

5 Choix techniques

5.1 Composants Standalone Angular

Angular 19 introduit les composants standalone, ce qui permet d'éviter l'utilisation excessive de `NgModules`, réduisant ainsi la complexité du code et accélérant le chargement des pages.

5.2 Reactive Forms

Les **Reactive Forms** ont été privilégiés pour leur flexibilité dans la gestion dynamique des formulaires. Par exemple, lors de l'édition de projets ou de tâches, il est facile de mettre à jour les valeurs de formulaire via `patchValue`. De plus, la gestion des validations complexes est plus explicite.

5.3 RxJS

L'utilisation de **RxJS** permet de gérer de manière réactive les flux de données via des `Observable`, tout en offrant des solutions élégantes pour gérer les appels asynchrones, la composition de données et les événements en temps réel (notifications).

5.4 Sécurité Front

L'utilisation de **JWT** pour l'authentification et la gestion des rôles est implémentée via un `AuthGuard` pour protéger l'accès aux différentes routes en fonction des rôles.

5.5 Drag & Drop

Angular CDK a été utilisé pour la fonctionnalité de **drag & drop** dans la gestion des réunions, ce qui permet d'avoir une solution native à Angular sans dépendre d'une bibliothèque externe.

5.6 DOM sécurisé

Le contrôle du DOM est effectué via `Renderer2`, ce qui permet de respecter les bonnes pratiques Angular et d'éviter les accès directs au DOM.

6 Captures d'écran

La section suivante présente différentes captures d'écran illustrant les principales fonctionnalités de l'application, permettant de visualiser son interface et son fonctionnement.

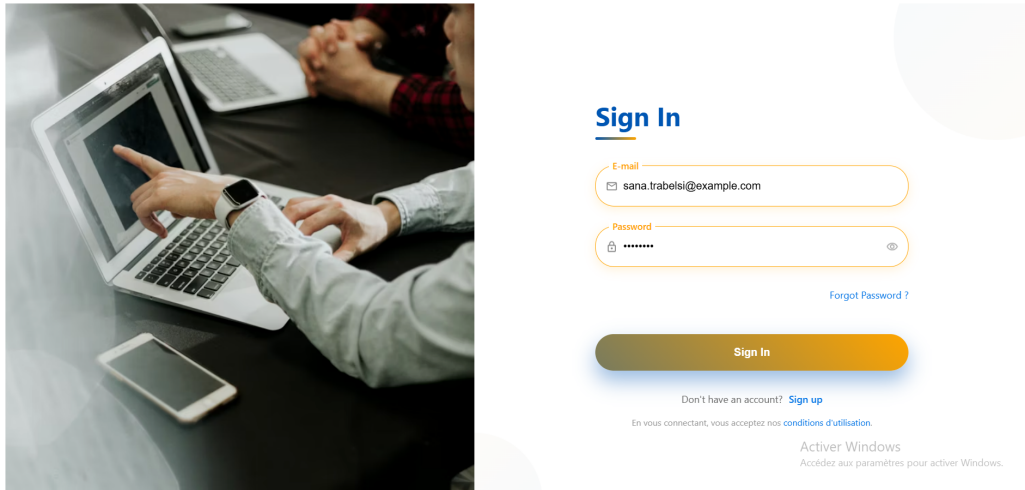


Figure 1: Page de connexion avec animation des particules

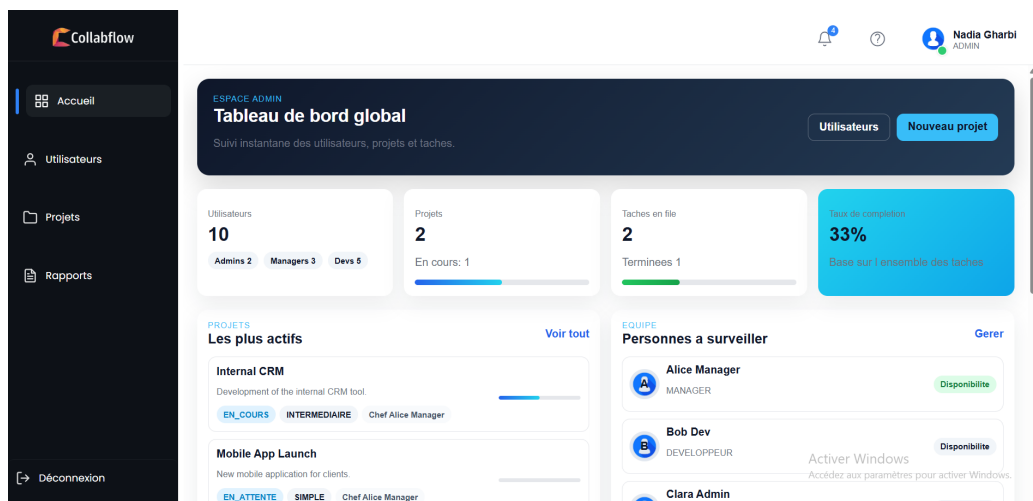


Figure 2: Tableau de bord Admin avec les cartes d'indicateurs

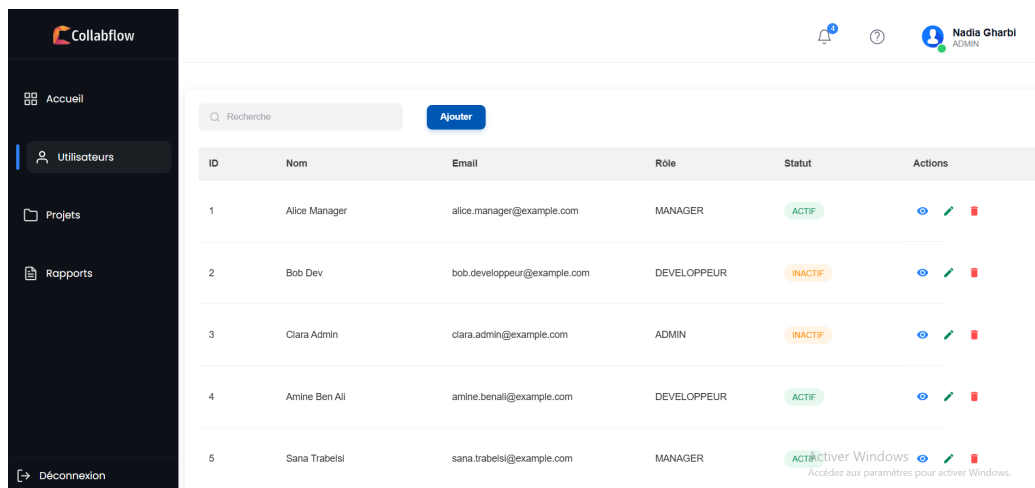


Figure 3: Vue de la gestion des utilisateurs avec filtres et modals

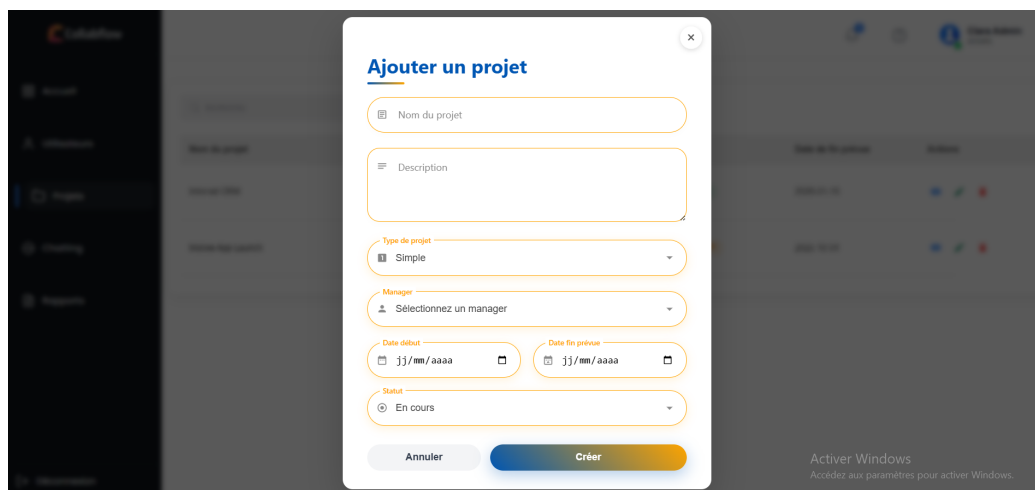


Figure 4: Vue de la gestion des projets (ajout d'un projet)

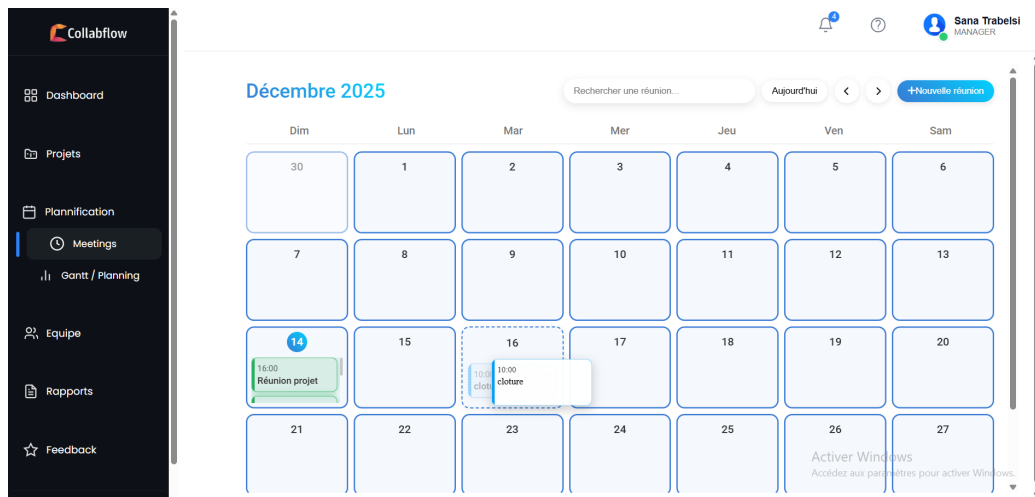


Figure 5: Calendrier des reunions avec drag and drop

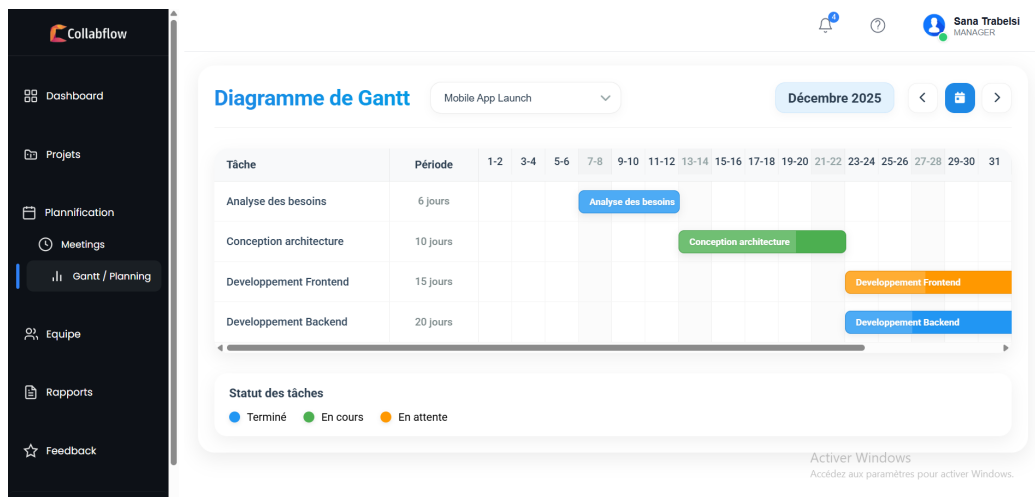


Figure 6: Diagramme de Gantt avec fonctionnalité de drag & drop

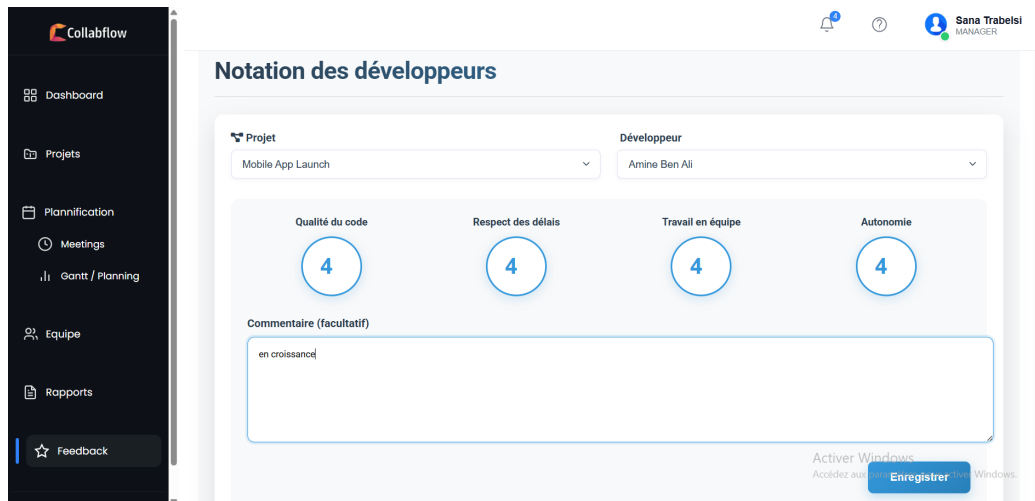


Figure 7: Vue des feedbacks

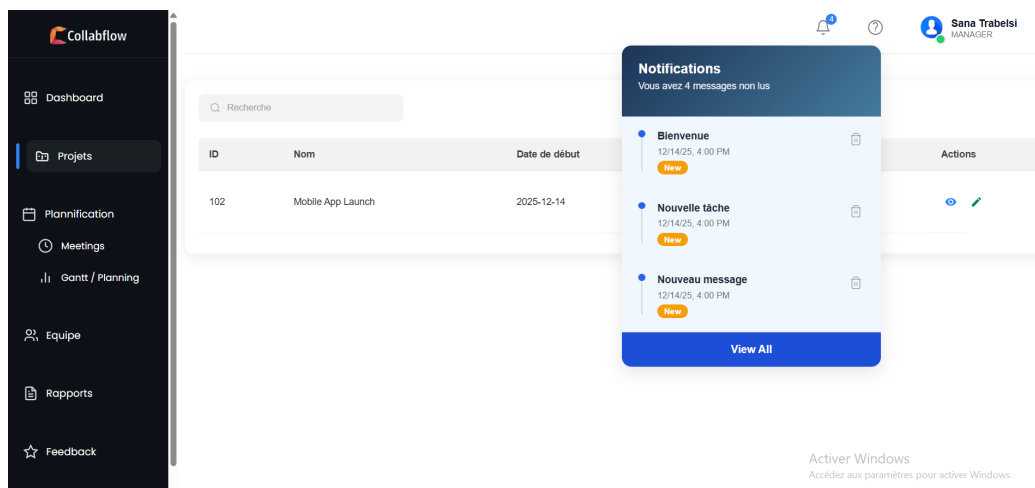


Figure 8: Vue des notifications

7 Conclusion et limites

Ce projet présente une application de gestion de projet front-end fonctionnelle avec une structure bien définie. Cependant, certaines limitations existent, telles que l'absence de persistance des données (données simulées) et la difficulté de tester certaines fonctionnalités dans des conditions réelles. Il serait nécessaire de travailler sur la gestion de l'intégration et des interactions avec des systèmes externes dans les prochaines étapes.

7.1 Pistes d'amélioration

L'une des pistes d'amélioration principales serait d'intégrer un système permettant la gestion réelle des données et d'améliorer la persistance ainsi que la gestion des interactions en temps réel au sein de l'application.