

Master Systèmes intelligents pour L'éducation

Module : [Développement web et mobile]

RAPPORT DE Mini Projet

De Fin De Module

Réalisé par :

Bouba Ahmed

Encadré par :

Pr. BENABBES KHALID

Année Universitaire 2024-2025

Table des matières

Résumé Exécutif	3
1 - Introduction	4
1.1 Contexte du Projet	4
1.2 Objectifs Spécifiques du Projet	4
1.3 Technologies Utilisées	5
2 - Description du Projet et des Fonctionnalités	6
2.1 Description du Projet	6
2.2 Fonctionnalités Principales	6
2.2.1 Gestion des Utilisateurs	6
2.2.2 Gestion des Projets	6
2.2.3 Gestion des Tâches	6
2.2.4 Gestion des Relations entre Données	7
2.2.5 Interface Utilisateur	7
2.2.6 Productivité	7
3 - Processus de Développement	8
3.1 Installation et Configuration de Laravel et Breeze	8
3.2 Ajouter Inertia JS	8
3.3 Ajout de React.js	9
3.4 Gestion des Routes et des Composants	9
3.5 La Base de Données	10
3.5.1 Tables	10
3.5.2 Relations	11
3.5.3 Déploiement	11
3.6 Tests et Validation	12
3.6.1 Résultats de test	12
4 Résultats	14
4.1 Fonctionnalités Développées	14
4.2 Illustrations des Pages Principales	14
5 Conclusion	18
5.1 Bilan du Projet	18
5.2 Difficultés Rencontrées et Solutions Apportées	18

Annexes	19
5.3 Code Source	19
5.4 Liens vers des Ressources ou Tutoriels Utilisés	19
5.5 Lien vers le Dépôt Git	19

Résumé Exécutif

Le projet de gestion des tâches vise à offrir une solution innovante et complète pour la gestion des projets et des tâches au sein des équipes. Ce système permet aux utilisateurs de suivre et gérer efficacement leurs tâches tout au long de leur cycle de vie, depuis leur création jusqu'à leur achèvement. En centralisant les informations relatives aux projets, aux tâches et aux affectations des utilisateurs, le système facilite la communication et la collaboration entre les membres d'une équipe, tout en garantissant une meilleure organisation et un suivi précis de l'avancement.

L'outil permet de gérer plusieurs projets simultanément, chacun avec ses propres tâches et affectations. Il offre une vue d'ensemble du statut des projets et des tâches, avec des fonctionnalités de suivi en temps réel, permettant aux utilisateurs de visualiser le pourcentage d'achèvement des différentes tâches et projets. De plus, le système offre des fonctionnalités pour l'affectation des tâches, la gestion des priorités, et le suivi des retards, afin d'assurer que les projets respectent les délais et objectifs définis.

En termes d'impact, le projet permet de renforcer la productivité et l'efficacité des équipes de travail, en offrant une interface intuitive et facile à utiliser pour la gestion des projets.

1 - Introduction

1.1 Contexte du Projet

Les entreprises et organisations font face à des défis croissants en matière de gestion de projets, nécessitant une coordination efficace des équipes et une gestion rigoureuse des délais et des ressources. Les outils existants, souvent complexes et coûteux, ne répondent pas toujours aux besoins des petites et moyennes structures.

Ce projet vise à offrir une solution simple et accessible, permettant de centraliser la gestion des projets et des tâches. Il facilite la collaboration, améliore la visibilité sur l'avancement des projets, et aide à prendre des décisions éclairées pour assurer la réussite des projets dans les délais impartis.

1.2 Objectifs Spécifiques du Projet

Le projet vise à répondre à plusieurs objectifs spécifiques afin de garantir une solution robuste, performante et adaptée aux besoins identifiés. Les objectifs incluent :

- ★ **Mise en place d'une plateforme dynamique** : Créer une application web moderne permettant une gestion fluide des projets et des tâches associées, tout en assurant une expérience utilisateur optimale grâce à l'intégration de React.js et Inertia.js.
- ★ **Gestion efficace des utilisateurs et de leurs rôles** : Mettre en œuvre un système d'authentification sécurisé avec Laravel Breeze, offrant une gestion différenciée pour les administrateurs et les membres réguliers.
- ★ **Structuration des données** : Concevoir et implémenter une base de données relationnelle cohérente avec des relations bien définies entre utilisateurs, projets, tâches, et assignations.
- ★ **Modularité et réutilisabilité du code** : Utiliser des composants React modulaires pour maximiser la réutilisation du code dans les différentes parties de l'application.
- ★ **Expérience utilisateur réactive** : Garantir une navigation fluide et sans rechargement de page en tirant parti de la puissance combinée d'Inertia.js et React.js.
- ★ **Personnalisation et évolutivité** : Offrir une interface utilisateur personnalisable avec Tailwind CSS et un backend Laravel facilement extensible pour accueillir des fonctionnalités futures.
- ★ **Suivi et gestion des projets** : Permettre aux utilisateurs de suivre facilement l'état d'avancement de leurs projets et tâches à travers des interfaces intuitives.

1.3 Technologies Utilisées

Le développement de l'application repose sur un ensemble de technologies modernes qui assurent performance, scalabilité, et maintenabilité. Voici les technologies principales employées dans le projet :

- ★ **Laravel** : Framework PHP backend utilisé pour gérer le routage, les contrôleurs, les modèles, et l'API REST. Il offre des fonctionnalités intégrées pour l'authentification et la gestion des bases de données.
- ★ **Breeze** : Un starter kit d'authentification léger et simple, intégré à Laravel, pour fournir rapidement une gestion des utilisateurs et des sessions.
- ★ **Inertia.js** : Une bibliothèque qui connecte Laravel et React, permettant de construire des applications SPA (Single Page Applications) tout en conservant une structure côté serveur.
- ★ **React.js** : Bibliothèque JavaScript utilisée pour concevoir des interfaces utilisateur dynamiques et réactives. Elle structure l'application en composants modulaires et réutilisables.
- ★ **Tailwind CSS** : Framework CSS qui permet de concevoir rapidement des interfaces utilisateur esthétiques et responsives à l'aide de classes utilitaires.
- ★ **MySQL** : Système de gestion de bases de données relationnelles choisi pour le stockage des données structurées. Il est fiable, performant, et bien intégré avec Laravel.
- ★ **Git et GitHub** : Outils de contrôle de version et plateforme de collaboration pour le suivi des modifications et la gestion du code source.
- ★ **Node.js et npm** : Utilisés pour gérer les dépendances frontend et compiler les ressources via des outils comme Vite.

2 - Description du Projet et des Fonctionnalités

2.1 Description du Projet

Ce projet est une application web développée pour simplifier la gestion des utilisateurs, des projets, et des tâches. Il permet aux administrateurs et aux membres de collaborer efficacement en suivant les progrès, en assignant des responsabilités, et en surveillant les échéances. Grâce à une interface utilisateur moderne et réactive, alimentée par React.js et Laravel, l'application garantit une expérience utilisateur fluide et une gestion centralisée des données.

2.2 Fonctionnalités Principales

Le projet intègre plusieurs fonctionnalités conçues pour faciliter la gestion des utilisateurs, des projets et des tâches. Ces fonctionnalités sont réparties selon les catégories suivantes :

2.2.1 Gestion des Utilisateurs

- **Authentification et Autorisation** : Inscription, connexion, réinitialisation de mot de passe et gestion des rôles (administrateur ou membre).
- **Profil Utilisateur** : Mise à jour des informations personnelles et changement de mot de passe.

2.2.2 Gestion des Projets

- **Création et Modification** : Ajout de nouveaux projets avec des détails tels que le nom, la description, les dates de début et de fin.
- **Assignment des Projets** : Attribution de projets aux membres pour assurer une meilleure organisation.
- **Suivi des Progrès** : Affichage des projets en cours, terminés ou en attente.

2.2.3 Gestion des Tâches

- **Ajout et Planification** : Création de tâches avec des priorités, des descriptions et des dates limites.
- **Assignment des Tâches** : Attribution des tâches à des utilisateurs spécifiques.

- **Mise à Jour du Statut** : Modification du statut des tâches (en cours, terminée, en attente).

2.2.4 Gestion des Relations entre Données

- **Hiérarchie des Projets et Tâches** : Chaque projet peut contenir plusieurs tâches, et chaque tâche est liée à un utilisateur assigné.
- **Rapports** : Génération de listes et de rapports sur les projets et les tâches assignées à un utilisateur ou un projet spécifique.

2.2.5 Interface Utilisateur

- **Tableaux de Bord** : Affichage centralisé des projets, des tâches, et des utilisateurs.
- **Expérience Interactive** : Navigation fluide grâce à Inertia.js et React.js, sans rechargement complet de la page.
- **Design Réactif** : Mise en page optimisée pour les écrans de différentes tailles grâce à Tailwind CSS.

2.2.6 Productivité

- **Recherche et Filtres** : Recherche avancée et filtres pour localiser rapidement des projets ou des tâches spécifiques.

3 - Processus de Développement

3.1 Installation et Configuration de Laravel et Breeze

Pour démarrer mon projet, j'ai utilisé Laravel, un framework PHP très prisé pour le développement d'applications web. La première étape a été d'installer Laravel via Composer. Ensuite, j'ai intégré Laravel Breeze pour gérer l'authentification de manière simple. Breeze offre une mise en place rapide avec des vues Blade et une configuration par défaut pour l'enregistrement, la connexion et la réinitialisation du mot de passe. Pour installer Breeze, j'ai utilisé la commande suivante :

```
1 composer create-project laravel/laravel task-manager
2 composer require laravel/breeze --dev
3 php artisan breeze:install
```

Compilation des Actifs Frontend

Pour compiler les actifs frontend, j'ai utilisé les commandes suivantes :

```
1 npm install && npm run dev
```

Cela permet de bénéficier d'un système d'authentification léger et rapide pour les utilisateurs de l'application.

3.2 Ajouter Inertia JS

J'ai ajouté Inertia.js pour offrir une expérience utilisateur moderne en utilisant des pages dynamiques côté client, tout en conservant Laravel comme serveur. Cette approche me permet d'utiliser des vues React sans renoncer aux avantages du routage traditionnel de Laravel. Inertia agit comme un véritable "pont" entre le serveur Laravel et le client React, ce qui me permet de rendre des pages sans avoir à recharger le navigateur. Pour installer Inertia.js, j'ai utilisé la commande suivante :

```
1 composer require inertiajs/inertia-laravel
2 npm install @inertiajs/inertia @inertiajs/inertia-react
```

Configuration de Fichier App.jsx :

```
1 const appName = import.meta.env.VITE_APP_NAME || 'Laravel';
2
3 createInertiaApp({
4   title: (title) => `${title} - ${appName}`,
5   resolve: (name) =>
```

```

6      resolvePageComponent (
7          './Pages/${name}.jsx',
8          import.meta.glob('./Pages/**/*.jsx'),
9      ),
10     setup({ el, App, props }) {
11         const root = createRoot(el);
12
13         root.render(<App {...props} />);
14     },
15     progress: {
16         color: '#4B5563',
17     },
18 });

```

3.3 Ajout de React.js

Pour la partie frontend, j'ai intégré React.js à mon projet afin de créer des interfaces utilisateur réactives et interactives. Après avoir configuré Inertia.js, j'ai installé React ainsi que ses dépendances nécessaires. React me permet de structurer l'interface utilisateur en composants, ce qui rend le code plus modulaire et réutilisable. L'intégration de React avec Inertia me permet de gérer les pages de manière dynamique tout en profitant des fonctionnalités de Laravel sur le backend. Voici comment j'ai installé React :

```
1 npm install react react-dom
```

Une fois cette installation terminée, nous avons créé les composants React qui interagiront avec les routes et les actions Laravel.

3.4 Gestion des Routes et des Composants

Laravel gère le routage des requêtes HTTP, tandis que j'utilise Inertia.js et React.js pour la gestion des vues côté client. Les routes Laravel définissent les points d'entrée pour les différentes pages, et chaque route peut renvoyer une vue Inertia qui sera rendue avec React. Par exemple, pour afficher la page d'accueil de mon projet, j'ai défini une route Laravel qui est reliée à un composant React via Inertia. Ce processus me permet de rendre le frontend dynamique tout en conservant les avantages du routage et du contrôle serveur. Les routes sont définies dans les fichiers web.php de Laravel, tandis que les composants React sont stockés dans le répertoire ressources/js/composants.

Exemple de route :

```

1 Route::get('/dashboard', function () {
2     return Inertia::render('Dashboard');
3 })->middleware(['auth', 'verified'])->name('dashboard');

```

Exemple de contrôleur :

```

1 class ProjectController extends Controller
2 {

```

```

3
4   public function __construct()
5   {
6       $this->middleware('isSuperuser')->except('index');
7   }
8
9   public function index()
10  {
11      $projects = Project::with(['user', 'tasks'])->get();
12      return Inertia::render('Projects/Index', [
13          'projects' => $projects,
14      ]);
15  }

```

Chaque composant React correspond à une vue et peut être personnalisé pour afficher des données dynamiques ou interagir avec l'utilisateur de manière fluide.

3.5 La Base de Données

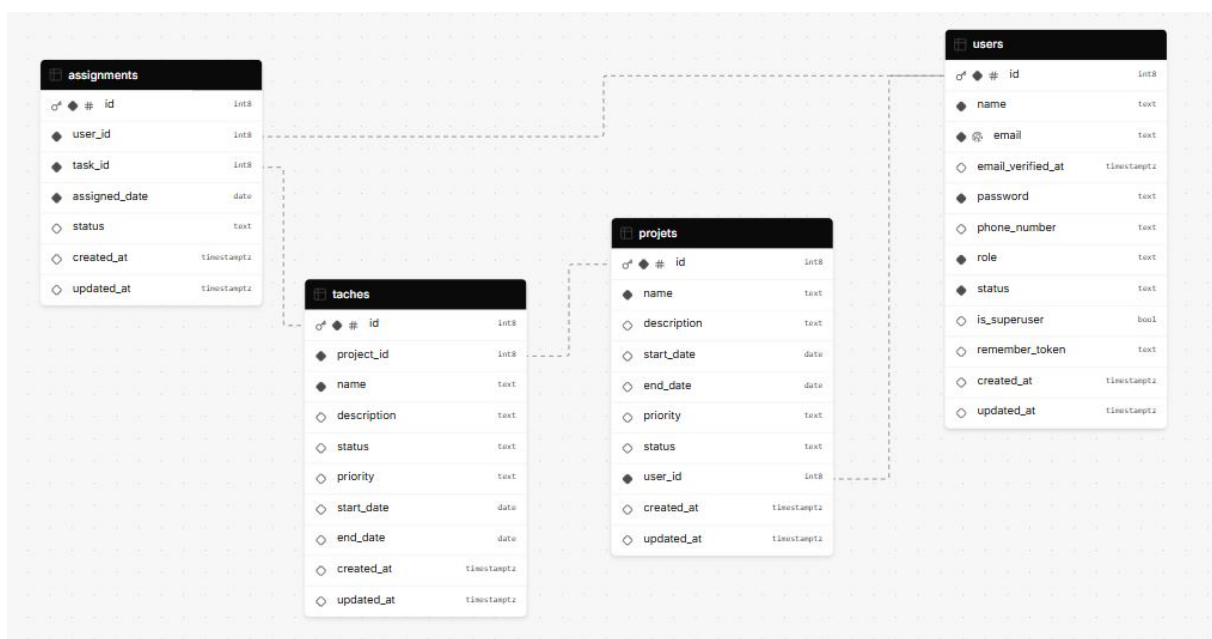


FIGURE 3.1 – Table users

3.5.1 Tables

- ✓ **Users** : Stocke les informations des utilisateurs, incluant leurs noms, e-mails, mots de passe, rôles et statuts.
- ✓ **Projects** : Gère les projets créés par les utilisateurs, avec des informations telles que le nom, la description, les dates de début et de fin, et le responsable.
- ✓ **Tasks** : Gère les tâches associées aux projets, avec des détails tels que le nom, la description, la priorité, le statut, et les dates de début et de fin.

- ✓ **Assignments** : Gère les affectations des utilisateurs aux tâches, incluant la date d'assignation et le statut.

3.5.2 Relations

- ✓ **Utilisateurs et Projets** : Chaque projet est associé à un utilisateur, indiqué par la clé étrangère user-id dans la table Projets.
- ✓ **Projets et Tâches** : Chaque tâche est associée à un projet, indiqué par la clé étrangère project-id dans la table Tâches.
- ✓ **Utilisateurs et Affectations** : Chaque affectation est associée à un utilisateur, indiqué par la clé étrangère user-id dans la table Assignments.
- ✓ **Tâches et Affectations** : Chaque affectation est associée à une tâche, indiqué par la clé étrangère task-id dans la table Assignments .

Configuration du Fichier .env

Enfin, j'ai vérifié que le fichier .env contenait les configurations nécessaires pour la base de données, le serveur mail, et d'autres paramètres essentiels.

```

1 APP_NAME=task-manager
2 APP_ENV=local
3 APP_KEY=base64:XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 # Configuration de la base de données
8 DB_CONNECTION=mysql
9 DB_HOST=127.0.0.1
10 DB_PORT=3306
11 DB_DATABASE=gestion_taches
12 DB_USERNAME=root
13 DB_PASSWORD=boubaahmed
14
15 MAIL_MAILER=smtp
16 MAIL_HOST=smtp.gmail.com
17 MAIL_PORT=587
18 MAIL_USERNAME=ahmedbouba383@gmail.com
19 MAIL_PASSWORD="iwte xxxx seej xxxx"
20 MAIL_ENCRYPTION=tls
21 MAIL_FROM_ADDRESS=ahmedbouba383@gmail.com
22 MAIL_FROM_NAME="${APP_NAME}"

```

3.5.3 Déploiement

J'ai ajouté l'application à un dépôt GitHub pour faciliter le suivi des versions. Aucune modification additionnelle n'a été nécessaire pour le déploiement en production. Voici les étapes que j'ai suivies pour ajouter le projet au dépôt GitHub :

```

1 git init
2
3 git add .

```

```
4
5 git commit -m "first commit"
6
7 git remote add origin
  <https://github.com/BoubaAhmed/Task-manager-with-Inertia.git>
8
9 git push -u origin main
```

3.6 Tests et Validation

Des tests unitaires et fonctionnels ont été réalisés pour vérifier la conformité des fonctionnalités développées avec les exigences initiales. Laravel propose un outil intégré pour écrire et exécuter des tests, utilisé comme suit :

```
1 php artisan test
```

Les tests couvrent :

- ✓ Les routes et contrôleurs pour assurer leur fonctionnement.
- ✓ Les modèles pour vérifier les relations et les règles de validation.
- ✓ Les composants React pour garantir une interface utilisateur fluide.

3.6.1 Résultats de test

```
1 PS C:\Users\hp\Desktop\Master_SI\Dev
   Web_Mobile\Laravel\my-laravel-app> php artisan test
2
3 PASS  Tests\Unit\ExampleTest
4     that true is true
5
6 PASS  Tests\Feature\Auth\AuthenticationTest
7     login screen can be rendered
8     users can authenticate using the login screen
9     users can not authenticate with invalid password
10    users can logout
11
12 PASS  Tests\Feature\Auth\EmailVerificationTest
13    email verification screen can be rendered
14    email can be verified
15    email is not verified with invalid hash
16
17 PASS  Tests\Feature\Auth>PasswordConfirmationTest
18    confirm password screen can be rendered
19    password can be confirmed
20    password is not confirmed with invalid password
21
22 PASS  Tests\Feature\Auth>PasswordResetTest
23    reset password link screen can be rendered
24    reset password link can be requested
25    reset password screen can be rendered
26    password can be reset with valid token
```

```
27
28 PASS Tests\Feature\Auth\PasswordUpdateTest
29     password can be updated
30     correct password must be provided to update password
31
32 PASS Tests\Feature\Auth\RegistrationTest
33     registration screen can be rendered
34     new users can register
35
36 PASS Tests\Feature\ExampleTest
37
38 PASS Tests\Feature\ProfileTest
39     profile page is displayed
40     profile information can be updated
41     email verification status is unchanged when the email
42     address is unchanged
43     user can delete their account
44     correct password must be provided to delete account
45
46 Tests:      25 passed (61 assertions)
47 Duration: 36.09s
```

Comme vous pouvez le constater dans ce code, tous les tests ont été exécutés avec succès. Cela démontre que toutes les fonctionnalités de l'application fonctionnent comme prévu et que le système est stable.

4 Résultats

4.1 Fonctionnalités Développées

Dans le cadre de ce projet, j'ai développées les fonctionnalités suivantes :

- ✓ **Gestion des utilisateurs** : Création, modification, suppression et authentification des utilisateurs avec différents rôles (administrateur, utilisateur standard, etc.).
- ✓ **Gestion des projets** : Ajout, modification et suivi des projets assignés aux utilisateurs.
- ✓ **Gestion des tâches** : Création et assignation de tâches aux projets, suivi de leur statut et des dates importantes.
- ✓ **Affectations** : Association des utilisateurs à des tâches spécifiques avec un suivi des statuts d'affectation.
- ✓ **Dashboard interactif** : Visualisation des projets, tâches et affectations via une interface utilisateur intuitive.
- ✓ **Notifications** : Système de notifications pour alerter les utilisateurs des mises à jour ou des deadlines.

Ces fonctionnalités ont été intégrées dans une interface utilisateur moderne et responsive, garantissant une expérience fluide et cohérente.

4.2 Illustrations des Pages Principales

Les captures d'écran ci-dessous montrent les pages principales de l'application :

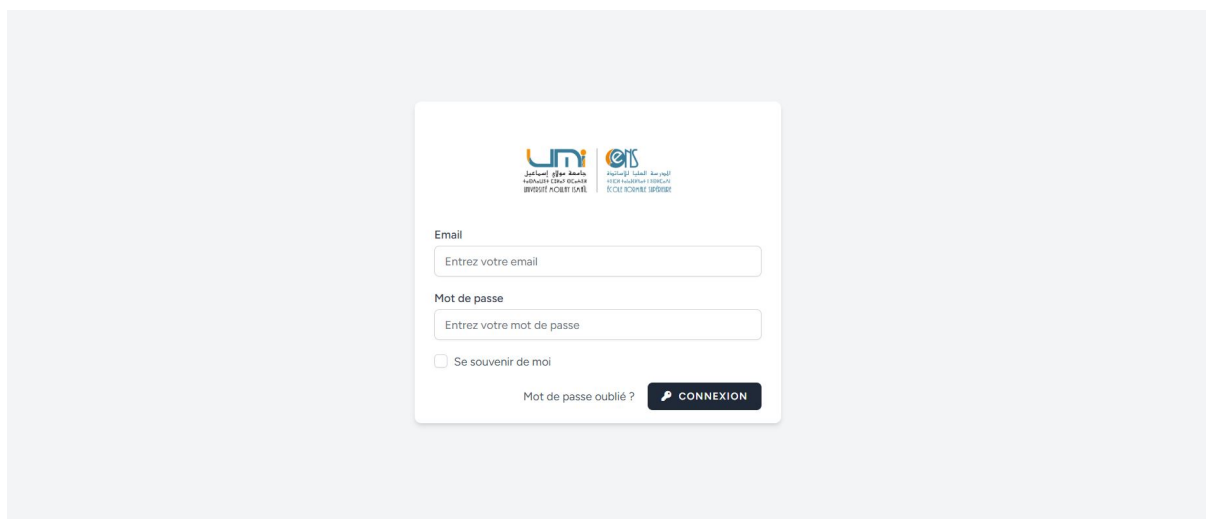


FIGURE 4.1 – Login

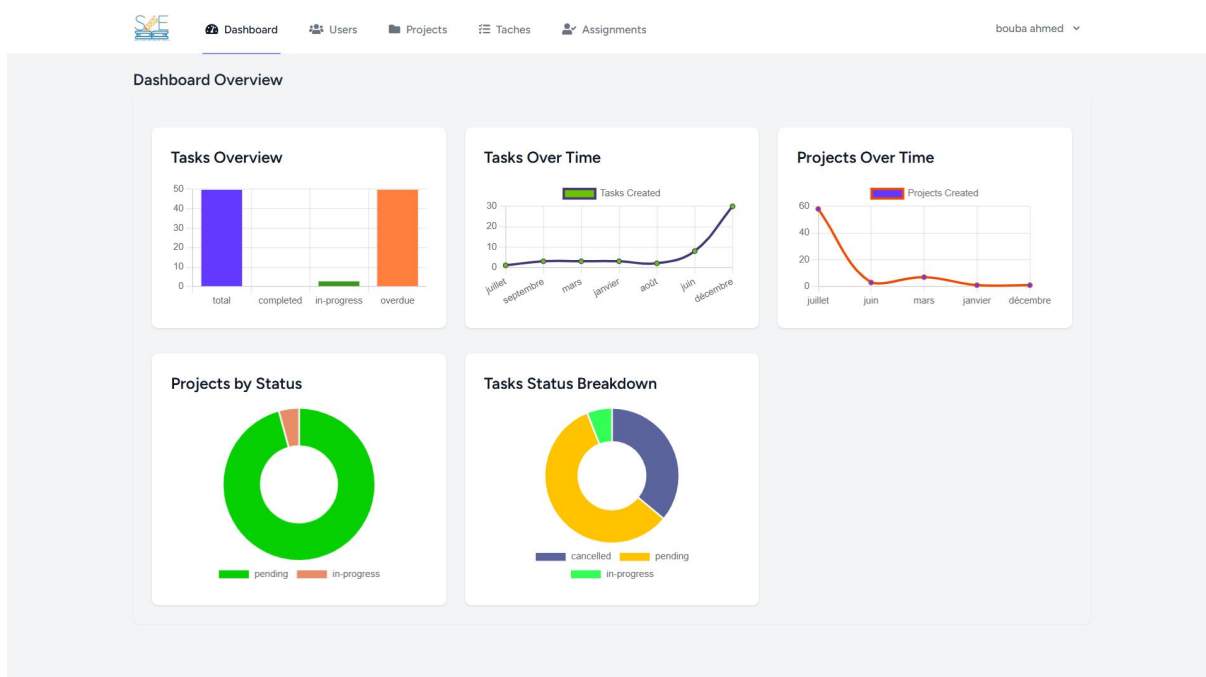
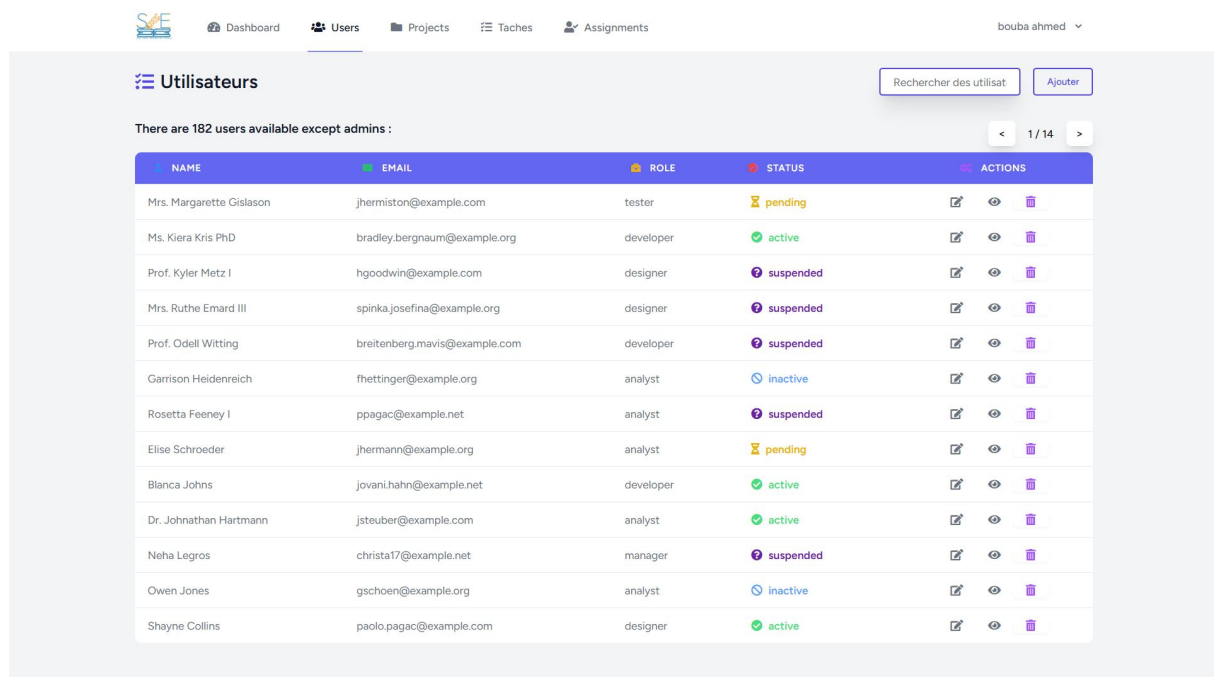


FIGURE 4.2 – Dashboard

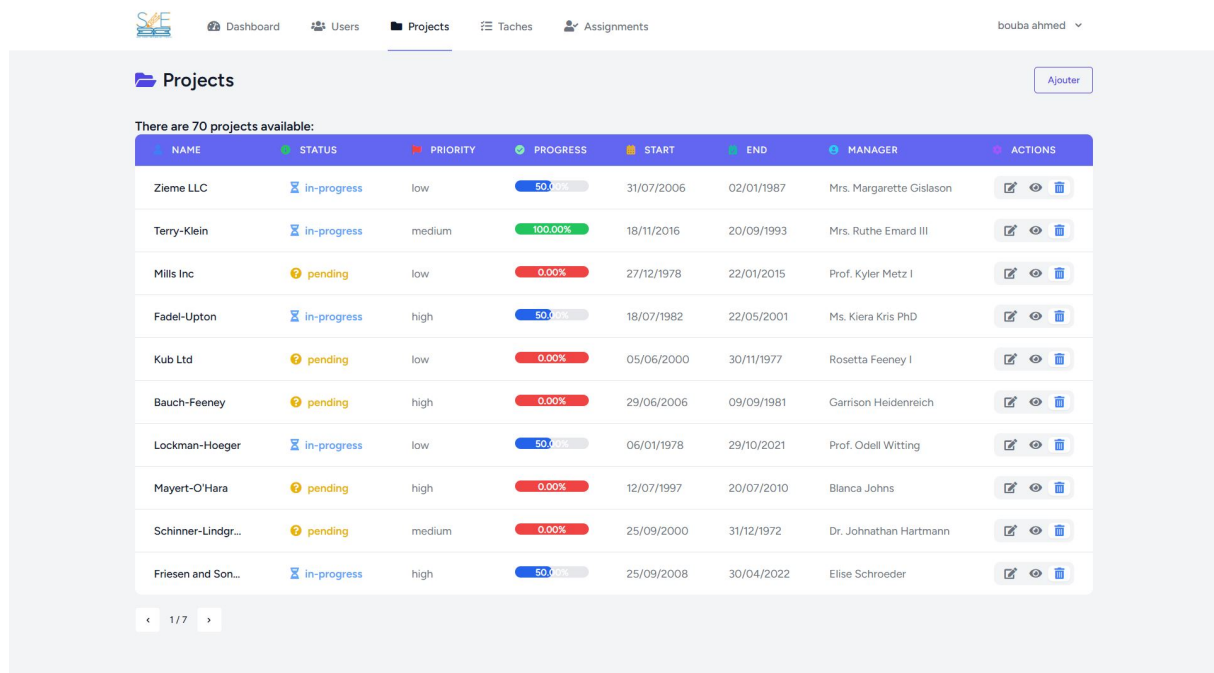


Utilisateurs

There are 182 users available except admins :

NAME	EMAIL	ROLE	STATUS	ACTIONS
Mrs. Margarette Gislason	jhermiston@example.com	tester	pending	
Ms. Kiera Kris PhD	bradley.berghaum@example.org	developer	active	
Prof. Kyler Metz I	hgoodwin@example.com	designer	suspended	
Mrs. Ruthe Emard III	spinka.josefina@example.org	designer	suspended	
Prof. Odell Witting	breitenberg.mavis@example.com	developer	suspended	
Garrison Heidenreich	fhettinger@example.org	analyst	inactive	
Rosetta Feeney I	ppagac@example.net	analyst	suspended	
Elise Schroeder	jhermann@example.org	analyst	pending	
Blanca Johns	jovani.hahn@example.net	developer	active	
Dr. Johnathan Hartmann	jsteuber@example.com	analyst	active	
Neha Legros	christa17@example.net	manager	suspended	
Owen Jones	gschoen@example.org	analyst	inactive	
Shayne Collins	paolo.pagac@example.com	designer	active	

FIGURE 4.3 – Page d'utilisateurs



Projets

There are 70 projects available:

NAME	STATUS	PRIORITY	PROGRESS	START	END	MANAGER	ACTIONS
Zieme LLC	in-progress	low	50.00%	31/07/2006	02/01/1987	Mrs. Margarette Gislason	
Terry-Klein	in-progress	medium	100.00%	18/11/2016	20/09/1993	Mrs. Ruthe Emard III	
Mills Inc	pending	low	0.00%	27/12/1978	22/01/2015	Prof. Kyler Metz I	
Fadel-Upton	in-progress	high	50.00%	18/07/1982	22/05/2001	Ms. Kiera Kris PhD	
Kub Ltd	pending	low	0.00%	05/06/2000	30/11/1977	Rosetta Feeney I	
Bauch-Feeney	pending	high	0.00%	29/06/2006	09/09/1981	Garrison Heidenreich	
Lockman-Hoeger	in-progress	low	50.00%	06/01/1978	29/10/2021	Prof. Odell Witting	
Mayert-O'Hara	pending	high	0.00%	12/07/1997	20/07/2010	Blanca Johns	
Schinner-Lindgr...	pending	medium	0.00%	25/09/2000	31/12/1972	Dr. Johnathan Hartmann	
Friesen and Son...	in-progress	high	50.00%	25/09/2008	30/04/2022	Elise Schroeder	

FIGURE 4.4 – Pages des Projets

Tâches

Rechercher des tâches | Filtrer par projet | Filtrer par statut | Ajouter | Mes Tâches

Il y a 50 tâches disponibles :

1 / 5

Status	Progress	Task Name	Description	Deadline
completed	100.00 %	Smith, Bartoletti an... vel	Rerum repudiandae itaque ducimus voluptatem quia a...	Limite : 05/01/2025
pending	0.00 %	Shields, Rosenbaum a... quisquam	Et rerum accusamus architecto ab consectetur.	Limite : 04/11/1989
in-progress	33.33 %	Keebler LLC quam	Assumenda rerum qui dolorum voluptate.	Limite : 31/08/2025
cancelled	0.00 %	Gulgowski-Wehner est	Deleniti dolores quia est quod.	Limite : 25/05/1978
pending	0.00 %	Heathcote-Skiles maxime	Animi autem est minus voluptas.	Limite : 03/09/1984
pending	0.00 %	Marks Inc repellat	Quis fugiat est saepe officiis aspernatur sunt.	Limite : 03/12/1996
in-progress	50.00 %	Langosh Ltd consequatur	Unde ut harum nemo quis corporis minima quasi.	Limite : 12/02/1994
cancelled	0.00 %	Tromp, Runolfsdottir... sapiente	Consequatur dolorem itaque voluptatem fugiat.	Limite : 20/06/1988
in-progress	0.00 %	Jast, Osinski and Ro... repellendus	Commodi dolores quas velit.	Limite : 14/02/2025
cancelled	0.00 %	Wyman-Graham consectetur	In porro voluptas qui porro impedit quos molestias...	Limite : 14/04/1985
cancelled	0.00 %	Barrows, Carter and ... ab	Quia quam corrupti dignissimos harum.	Limite : 21/05/1991
cancelled	0.00 %	Dooley, Ebert and Di... temporibus	Libero aut et ut sunt enim aliquam laborum.	Limite : 23/05/2020

FIGURE 4.5 – Pages des Taches

Assignments :

Filtrer par utilisateur | Filtrer par tâche | Filtrer par projet | Ajouter

There are 60 assignments available:

Rechercher des tâches

USER	User Role	Task	Priority	Task Status	Assign Status	Assigned Date	Actions
bouba ahmed	developer	vel	high	completed	completed	30/12/2024	[Edit] [View] [Delete]
bouba ahmed	developer	consequatur	high	in-progress	completed	30/12/2024	[Edit] [View] [Delete]
bouba ahmed	developer	alias	low	completed	completed	30/12/2024	[Edit] [View] [Delete]
bouba ahmed	developer	sapiente	high	in-progress	completed	30/12/2024	[Edit] [View] [Delete]
bouba ahmed	developer	accusantium	low	in-progress	completed	30/12/2024	[Edit] [View] [Delete]
Katheryn Macejkovic	developer	sequi	medium	in-progress	pending	30/12/2024	[Edit] [View] [Delete]
bouba ahmed	developer	sequi	medium	in-progress	completed	30/12/2024	[Edit] [View] [Delete]
bouba ahmed	developer	veritatis	medium	in-progress	completed	30/12/2024	[Edit] [View] [Delete]
bouba ahmed	developer	aperiam	high	in-progress	completed	30/12/2024	[Edit] [View] [Delete]

1 of 7

FIGURE 4.6 – Pages des Assignations

5 Conclusion

5.1 Bilan du Projet

Le projet a permis de développer une application de gestion collaborative intégrant des fonctionnalités essentielles pour la gestion des projets et des tâches. L'utilisation des technologies modernes comme Laravel, React et Tailwind CSS a rendu l'application performante et ergonomique. Les objectifs fixés ont été atteints, bien que certains aspects puissent être améliorés pour de futures versions.

5.2 Difficultés Rencontrées et Solutions Apportées

- ✓ **Problème de compatibilité des versions** : Certaines bibliothèques présentaient des incompatibilités. La solution a consisté à utiliser des versions stables et compatibles.
- ✓ **Gestion des relations complexes** : La modélisation des relations entre utilisateurs, projets et tâches a nécessité une planification approfondie. L'utilisation des migrations Laravel a facilité la création des schémas relationnels.
- ✓ **Performance du frontend** : Le rendu côté client avec React a parfois posé des problèmes de lenteur. L'optimisation des composants React et la réduction des requêtes ont permis de résoudre ce problème.
- ✓ **Manque de documentation initiale** : Une documentation technique insuffisante a ralenti le démarrage. Une meilleure organisation des ressources et des tutoriels a aidé à surmonter cet obstacle.

Annexes

5.3 Code Source

Le code source principal du projet est disponible dans le dépôt GitHub. Vous y trouverez l'intégralité du projet, y compris la structure Laravel, l'intégration de React.js et Inertia.js, ainsi que les fichiers de configuration et les dépendances.

5.4 Liens vers des Ressources ou Tutoriels Utilisés

Voici quelques ressources et tutoriels que nous avons utilisés pendant le développement de ce projet :

- ✓ **Documentation Laravel** : <https://laravel.com/docs>
- ✓ **Inertia.js Documentation** : <https://inertiajs.com>
- ✓ **React.js Documentation** : <https://reactjs.org/docs/getting-started.html>
- ✓ **Breeze pour Laravel** : <https://laravel.com/docs/8.x/starter-kits#laravel-breeze>

5.5 Lien vers le Dépôt Git

Le code complet du projet est disponible sur GitHub à l'adresse suivante :

- ◇ **Dépôt GitHub** : <https://github.com/BoubaAhmed/Task-manager-with-Inertia.git>