Documentation du projet de gestion de projet

1. Introduction

Ce projet est une application web permettant de gérer des projets, d'organiser les tâches associées, et de suivre leur progression. L'application offre une gestion des utilisateurs avec différents rôles (Administrateur, Chef de projet, Membre) et intègre des fonctionnalités comme la création, la modification et la suppression de projets et de tâches.

Objectifs

- Simplifier la gestion des projets et des équipes.
- Offrir une interface intuitive et des outils pour suivre la progression des tâches.
- Faciliter la collaboration entre les utilisateurs.

2. Architecture du système

Vue d'ensemble

L'architecture est organisée en trois couches principales :

1. Frontend :
- Langage : HTML, CSS, JavaScript
- Frameworks/Bibliothèques : Aucune spécifique (approche vanilla JS et CSS personnalisé).

2. Backend :
- Langage : PHP
- Fonctionnalités principales : Gestion des utilisateurs, des projets et des tâches via des scripts PHP.

3. Infrastructure :
- Conteneurisation : Docker pour une installation et un déploiement uniformes.
- Base de données : MySQL.

Diagramme d’architecture

```
[Utilisateur] --> [Frontend (HTML/CSS/JS)] --> [Backend (PHP)] --> [Base de données (MySQL)]
```

3. Conception logicielle

Structure de la base de données

- Utilisateur : Gestion des informations des utilisateurs (rôle, identifiants, etc.).
- Projet : Données sur les projets (nom, description, chef de projet, etc.).
- Tâche : Détails des tâches (titre, description, priorité, état, date limite, etc.).
- Commentaires : Détails sur les commentaires (contenu, date d'écriture, personne qui l'écrit et la tâche liée au commentaire).

Diagramme des classes UML

1. User : Classe pour manipuler les données utilisateur.
2. Project : Classe pour gérer les projets et leurs attributs.
3. Task : Classe pour créer et modifier des tâches.
4. Comments : Classe pour créer des commentaires.

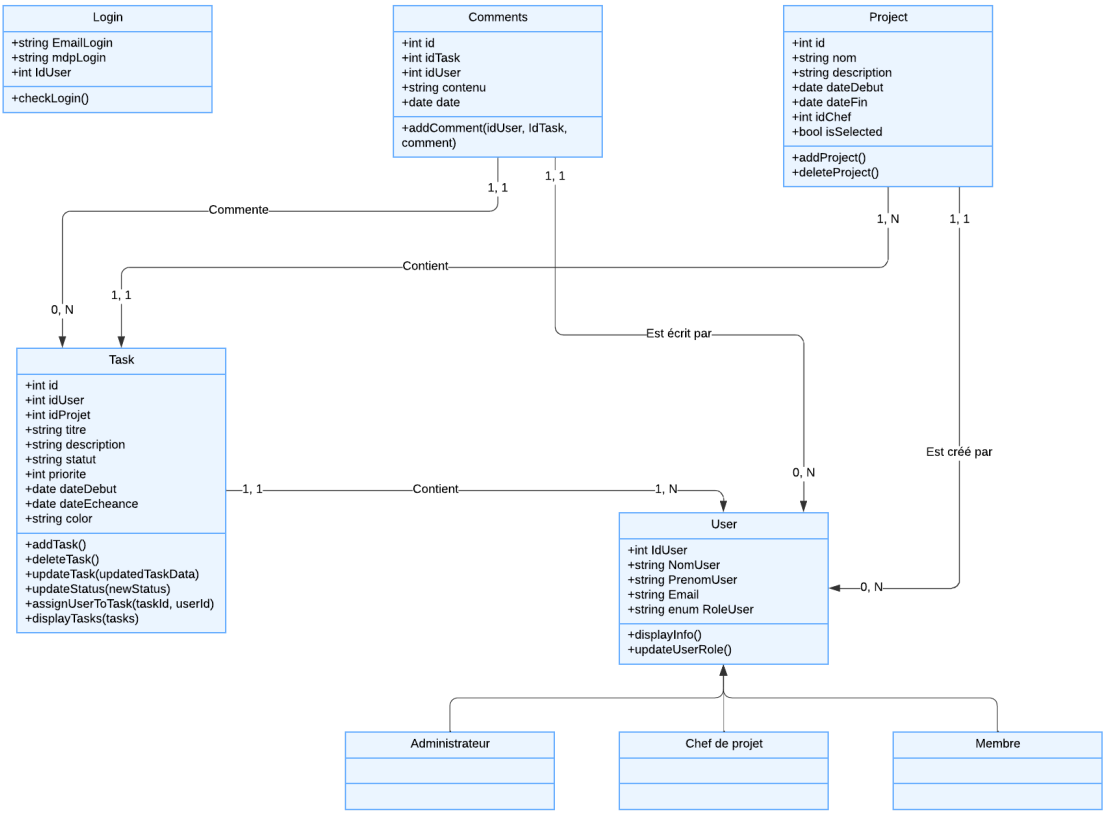
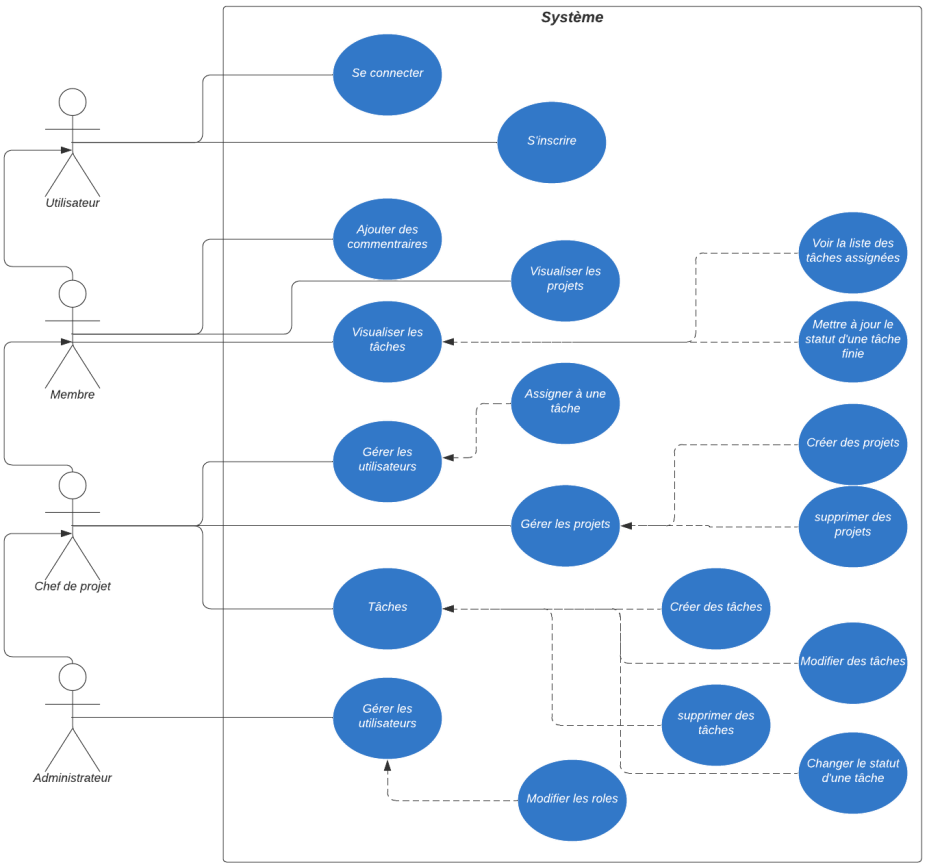


Diagramme de cas d’utilisation UML



Flux utilisateur

- Connexion : Les utilisateurs se connectent et accèdent à un tableau de bord personnalisé selon leur rôle. Ils peuvent soit créer un compte soit se connecter avec un compte existant.
- Gestion des projets : Les chefs de projet et les administrateurs créent et modifient des projets.
- Gestion des tâches : Les chefs de projet et les administrateurs créent et modifient des tâches. Les membres peuvent les consulter et les noter comme terminées pour mettre à jour la progression du projet.

La progression du projet peut être suivie grâce à l'indicateur de progression qui calcule le pourcentage de progression du projet en fonction du nombre de tâches finies sur le nombre total de tâches.

4. Mise en œuvre

Exemples de fonctionnalités implémentées

1. Ajout d'une tâche :

- Script : `addTask.php`
- Exemple de champ requis : `TitreTask`, `DescriptionTask`, `PrioriteTask`.
- Frontend : Formulaire accessible via un bouton « Ajouter une tâche ».

2. Gestion des permissions :

- Script :`checkPermissions.php`
- Fonction : Empêche les utilisateurs non autorisés de modifier certaines données.

3. WebSocket pour les indicateurs :

- Fichier : `dashboard/indicateur/server.js`
- Fonctionnalité : Permet une mise à jour en temps réel des statistiques sur le tableau de bord.

5. Tests et validation

Stratégie de tests

- Tests unitaires : Classes JavaScript (User, Project, Task).
- Tests fonctionnels : Interaction entre PHP et fonctions JavaScript dans les classes JavaScript.
- Tests d'intégration : Interaction entre PHP et MySQL.
- Tests manuels : Validation des fonctionnalités principales (ajout de tâches, modifications de projets).

Exécution des tests

- Lancer les tests unitaires avec Jest :

```
npm test
```

- Vérification manuelle :

Accédez à l'interface et testez les cas d'utilisation principaux.

6. Déploiement

Installation avec Docker :

1. Clonez le dépôt :

```
git clone <URL_DU_DEPOT>
cd Projet/docker
```

2. Lancez Docker Compose :

```
docker-compose up --build
```

3. Accédez à l'application via : `http://localhost:1234`.

7. Limites et pistes d'amélioration

- Limites actuelles :

- Pas de système de notifications pour les modifications des tâches.
- Gestion de la priorité des projets à améliorer.

- Pistes d'amélioration :

- Implémenter un système de chat interne pour faciliter la communication.
- Ajouter une API REST pour permettre l'intégration avec des outils externes.

8. Annexes

- Structure des dossiers : Voir la section correspondante du README.

- Fichier SQL : Disponible dans `gestion_projet.sql` pour initialiser la base de données.