

AMÉLIOREZ UNE APPLICATION EXISTANTE DE TODO & CO

PROJET 8

DÉVELOPPEUR D'APPLICATION - PHP / SYMFONY

Sommaire

1	Introduction	3
1.1	Contexte.....	3
2	Description du besoin	3
2.1	Corrections d'anomalies.....	3
2.1.1	Une tâche doit être attachée à un utilisateur	3
2.1.2	Choisir un rôle pour un utilisateur	3
2.2	Implémentation de nouvelles fonctionnalités	4
2.2.1	Autorisation	4
2.2.2	Implémentation de tests automatisés	4
2.2.3	Documentation technique	4
2.2.4	Audit de qualité du code & performance de l'application	4
3	Définition des acteurs	5
3.1	Acteurs principaux.....	5
3.2	Acteurs secondaires	5
4	Diagramme de cas d'utilisation	6
5	Diagrammes de séquence	7
5.1	Ajout d'un utilisateur (créer un compte)	7
5.2	Suppression d'un utilisateur	7
5.3	Mise à jour d'un utilisateur.....	8
5.4	Lister des utilisateurs	8
5.5	Lister les Tasks	9
5.6	Ajout d'un Task.....	9
5.7	Modification d'un Task	10
5.8	Suppression d'un Task.....	10
5.9	Modifier le statut d'un task.....	11
5.10	Authentification	11
6	Diagramme de classes	12
7	Modèle physique de données.....	13
8	Conclusion.....	14
9	Proposition d'améliorations.....	14
10	Annexe.....	14

1 Introduction

1.1 Contexte

Je viens d'intégrer une startup dont le cœur de métier est une application permettant de gérer ses tâches quotidiennes. L'entreprise vient tout juste d'être montée, et l'application a dû être développée à toute vitesse pour permettre de montrer à de potentiels investisseurs que le concept est viable (on parle de Minimum Viable Product ou MVP).

Mon rôle est d'améliorer la qualité de l'application. La qualité est un concept qui englobe bon nombre de sujets : on parle souvent de qualité de code, mais il y a également la qualité perçue par l'utilisateur de l'application ou encore la qualité perçue par les collaborateurs de l'entreprise, et enfin la qualité qu'on perçoit lorsqu'il nous faut travailler sur le projet.

Ainsi, pour ce dernier projet de spécialisation, vous êtes dans la peau d'un développeur expérimenté en charge des tâches suivantes :

- L'implémentation de nouvelles fonctionnalités ;
- La correction de quelques anomalies ;
- L'implémentation de tests automatisés.

Il vous est également demandé d'analyser le projet grâce à des outils vous permettant d'avoir une vision d'ensemble de la qualité du code et des différents axes de performance de l'application.

Il ne vous est pas demandé de corriger les points remontés par l'audit de qualité de code et de performance. Cela dit, si le temps vous le permet, ToDo & Co sera ravi que vous réduisiez la dette technique de cette application.

2 Description du besoin

2.1 Corrections d'anomalies

2.1.1 Une tâche doit être attachée à un utilisateur

Actuellement, lorsqu'une tâche est créée, elle n'est pas rattachée à un utilisateur. Il m'est demandé d'apporter les corrections nécessaires afin qu'automatiquement, à la sauvegarde de la tâche, l'utilisateur authentifié soit rattaché à la tâche nouvellement créée.

- Lors de la modification de la tâche, l'auteur ne peut pas être modifié.
- Pour les tâches déjà créées, il faut qu'elles soient rattachées à un utilisateur "anonyme".

2.1.2 Choisir un rôle pour un utilisateur

Lors de la création d'un utilisateur, il doit être possible de choisir un rôle pour celui-ci. Les rôles listés sont les suivants :

- Rôle utilisateur (*ROLE_USER*) ;
- Rôle administrateur (*ROLE_ADMIN*).

Lors de la modification d'un utilisateur, il est également possible de changer le rôle d'un utilisateur.

2.2 Implémentation de nouvelles fonctionnalités

2.2.1 Autorisation

- Seuls les utilisateurs ayant le rôle administrateur (*ROLE_ADMIN*) doivent pouvoir accéder aux pages de gestion des utilisateurs.
- Les tâches ne peuvent être supprimées que par les utilisateurs ayant créé les tâches en question.
- Les tâches rattachées à l'utilisateur "anonyme" peuvent être supprimées uniquement par les utilisateurs ayant le rôle administrateur (*ROLE_ADMIN*).

2.2.2 Implémentation de tests automatisés

Il m'est demandé d'implémenter les tests automatisés (tests unitaires et fonctionnels) nécessaires pour assurer que le fonctionnement de l'application est bien en adéquation avec les demandes.

Ces tests doivent être implémentés avec **PHPUnit** ; vous pouvez aussi utiliser Behat pour la partie fonctionnelle.

Je dois prévoir des données de tests afin de pouvoir prouver le fonctionnement dans les cas explicités dans ce document.

Il m'est demandé de fournir un rapport de couverture de code au terme du projet. Il faut que le taux de couverture soit supérieur à 70 %.

2.2.3 Documentation technique

Il vous est demandé de produire une documentation expliquant comment l'implémentation de l'authentification a été faite. Cette documentation se destine aux prochains développeurs juniors qui rejoindront l'équipe dans quelques semaines. Dans cette documentation, il doit être possible pour un débutant avec le framework Symfony de :

- Comprendre quel(s) fichier(s) il faut modifier et pourquoi ;
- Comment s'opère l'authentification ;
- Et où sont stockés les utilisateurs.

S'il me semble important de mentionner d'autres informations, je ne dois pas hésiter à le faire.

Par ailleurs, j'ouvre la marche en matière de collaboration à plusieurs sur ce projet. Il m'est également demandé de produire un document expliquant comment devront procéder tous les développeurs souhaitant apporter des modifications au projet.

Ce document devra aussi détailler le processus de qualité à utiliser ainsi que les règles à respecter.

2.2.4 Audit de qualité du code & performance de l'application

Les fondateurs souhaitent pérenniser le développement de l'application. Cela dit, ils souhaitent dans un premier temps faire un état des lieux de la dette technique de l'application.

Au terme de votre travail effectué sur l'application, il m'est demandé de produire un audit de code sur les deux axes suivants : la qualité de code et la performance.



Bien évidemment, il m'est fortement conseillé d'utiliser des outils me permettant d'avoir des métriques pour appuyer mes propos.

Concernant l'audit de performance, l'usage de Blackfire ou d'un autre profiler est obligatoire. Ce dernier nous permettra de produire des analyses précises et adaptées aux évolutions futures du projet.

3 Définition des acteurs

Dans le cadre de ma réflexion j'ai pu définir les acteurs suivants

3.1 Acteurs principaux

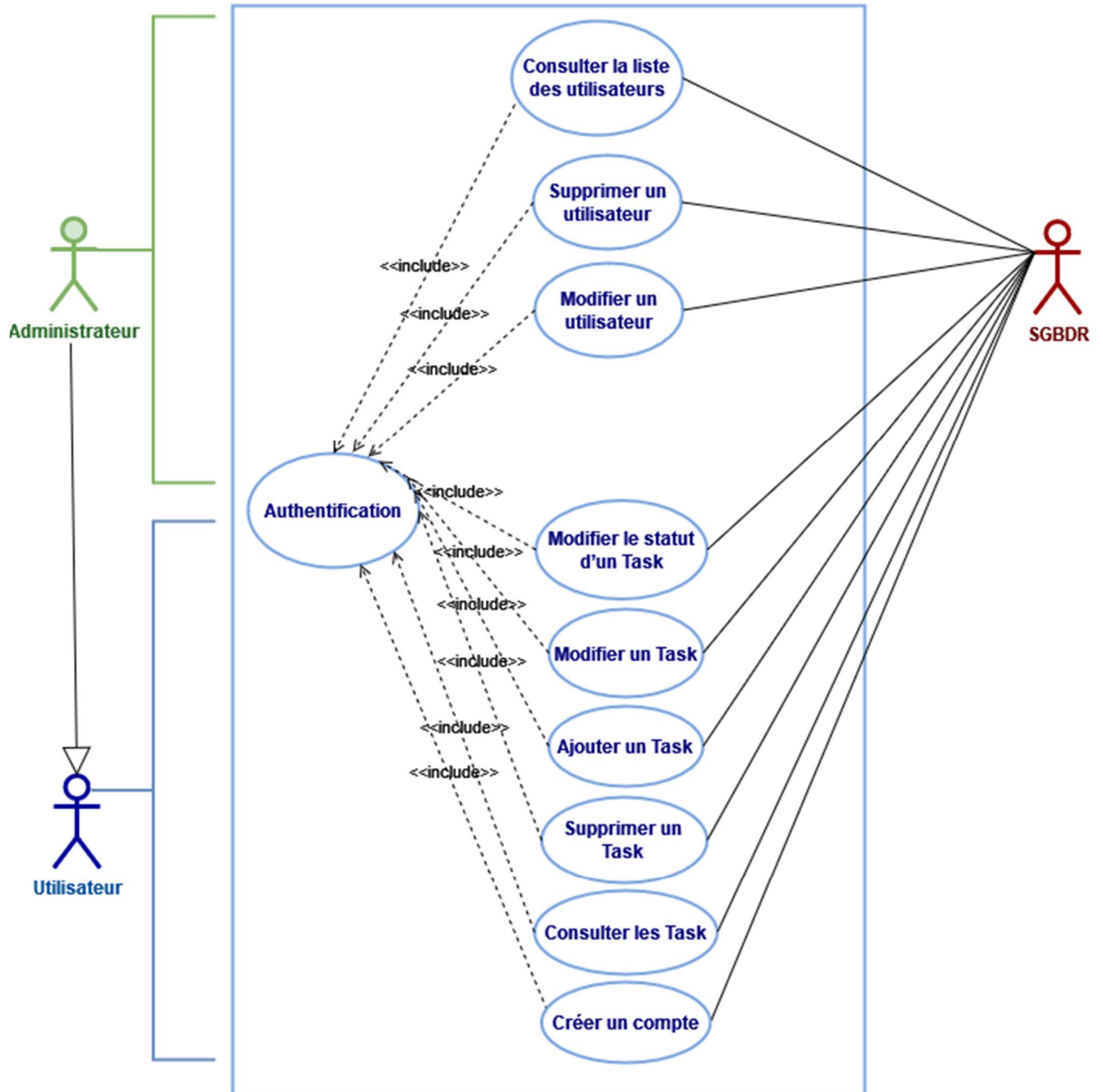
 Administrateur	Personne avec tous les privilèges
 Utilisateur	Personne avec des droits restreints

3.2 Acteurs secondaires



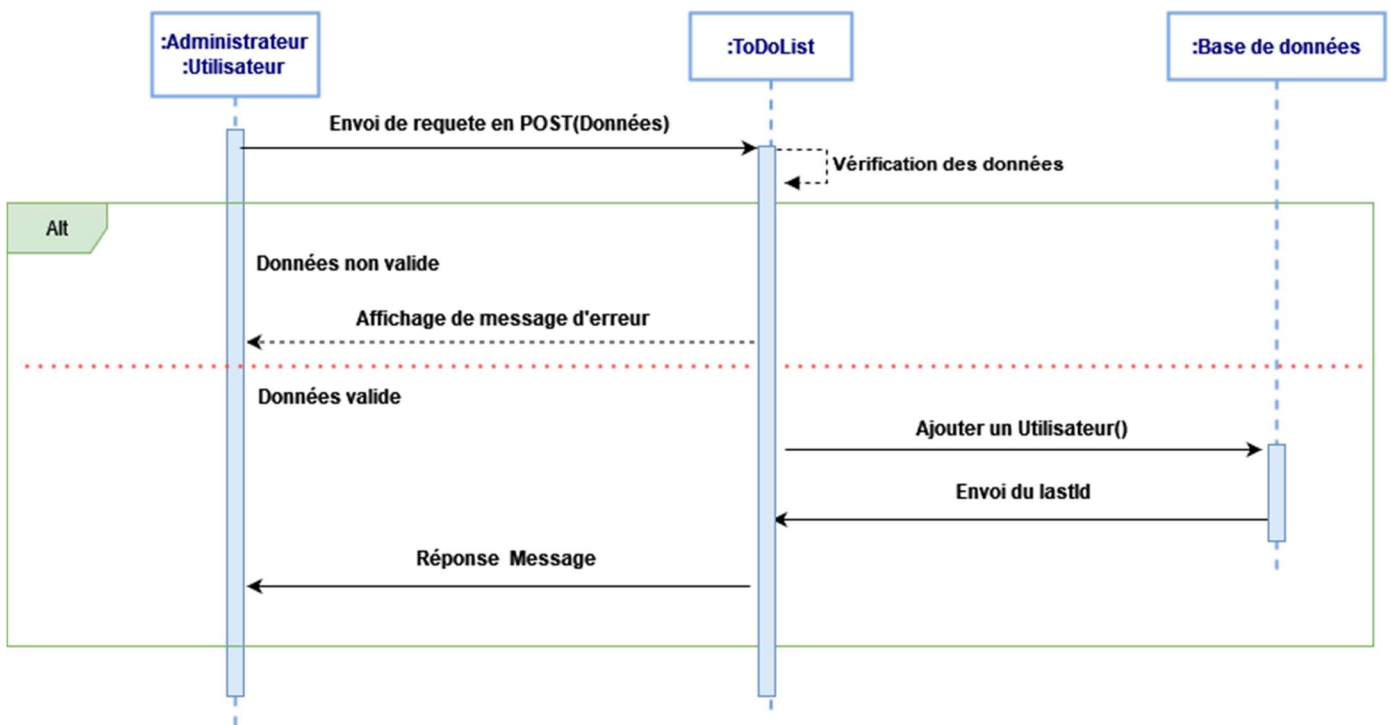
« Système » permettant la manipulation des données

4 Diagramme de cas d'utilisation

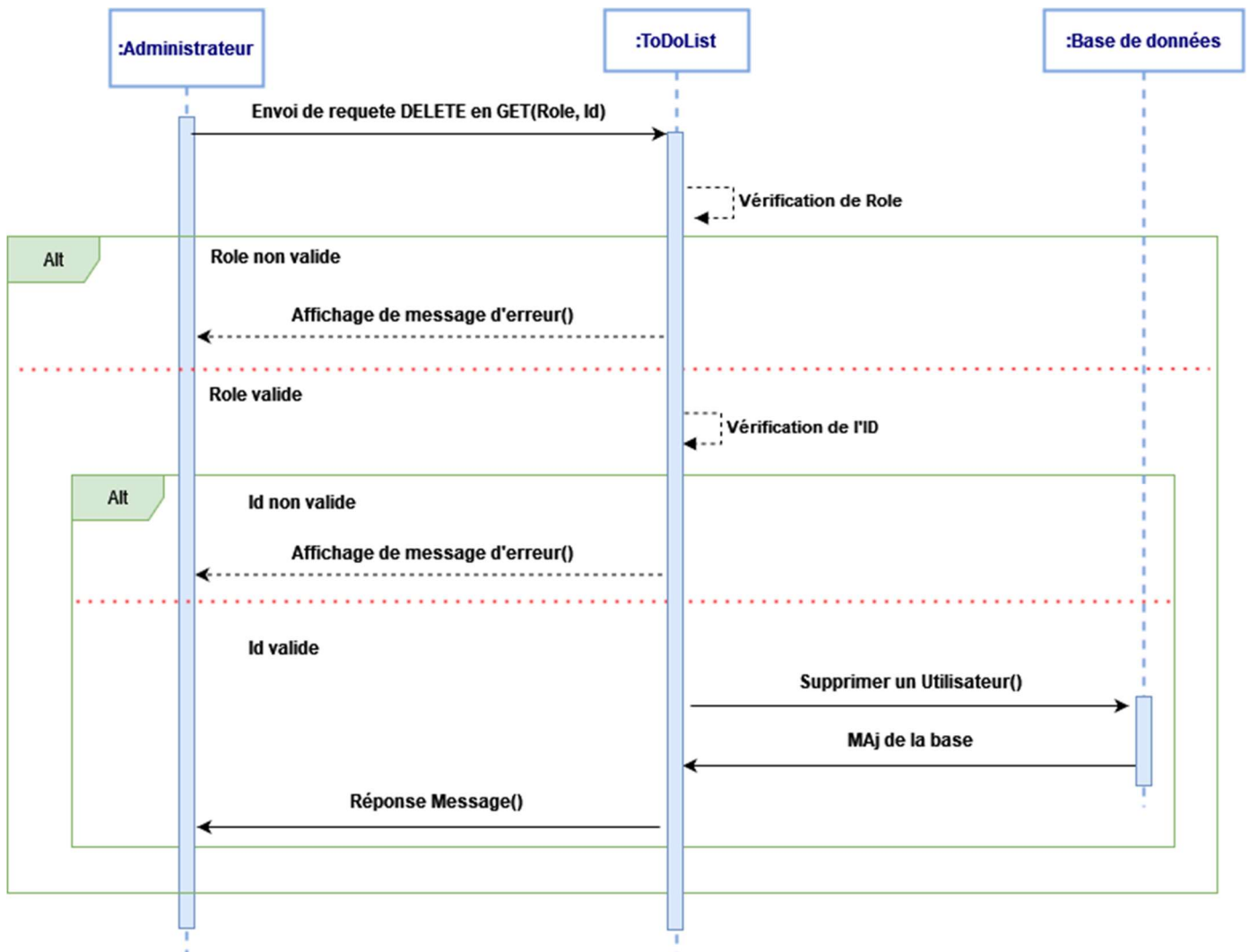


5 Diagrammes de séquence

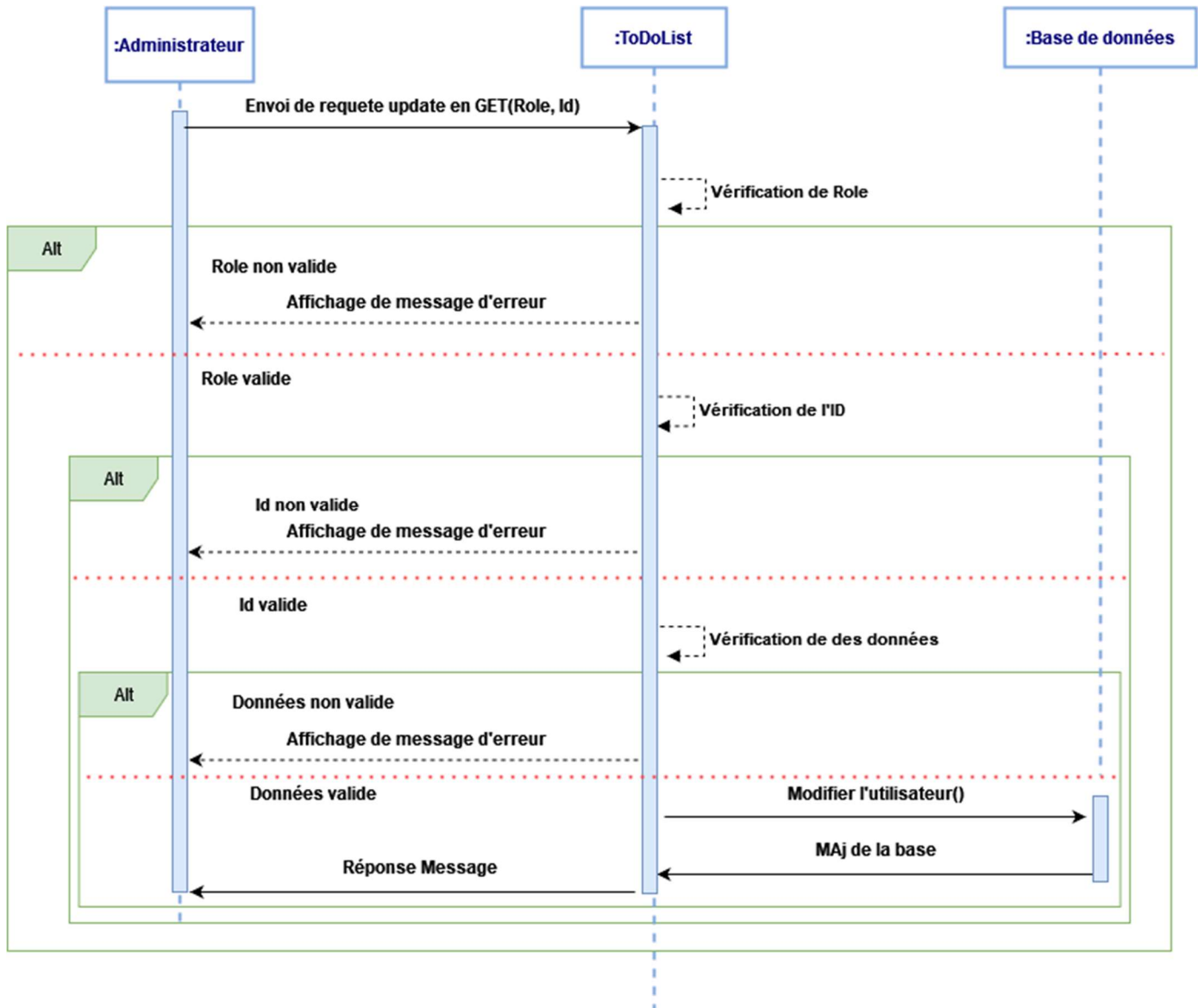
5.1 Ajout d'un utilisateur (créer un compte)



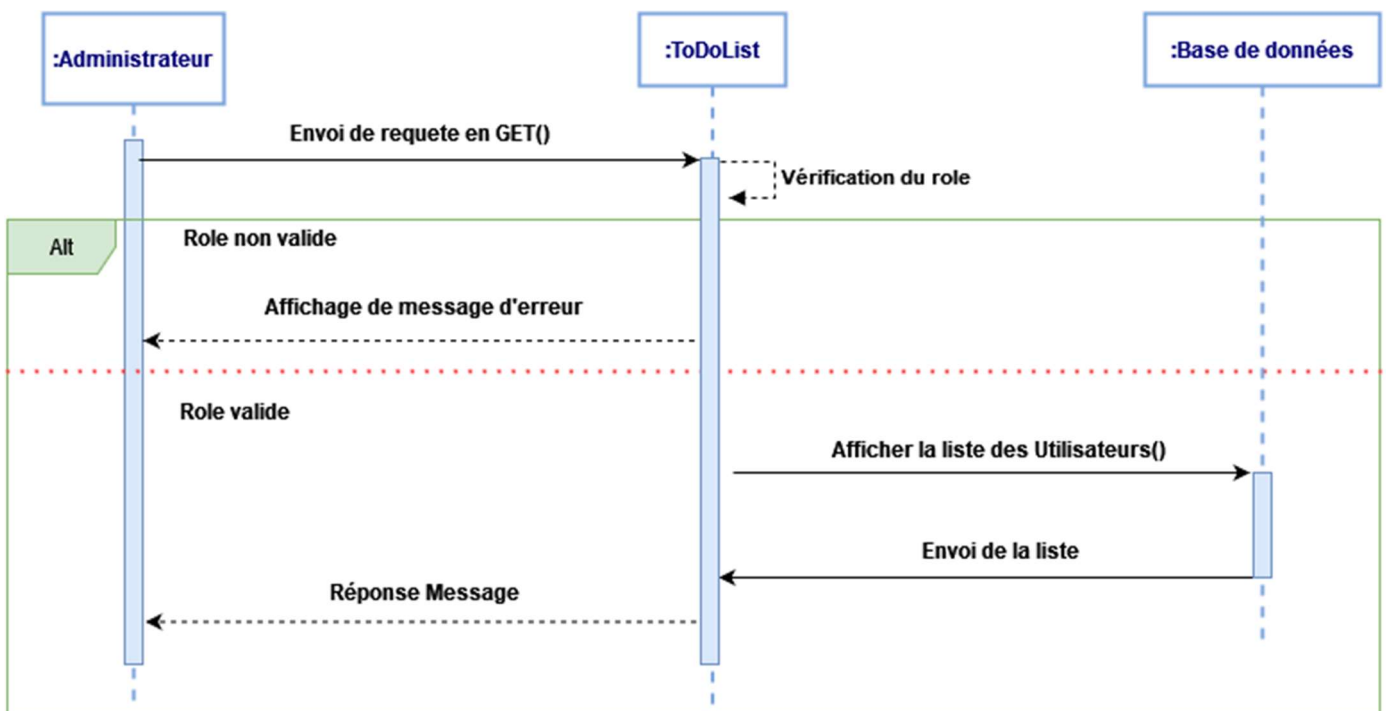
5.2 Suppression d'un utilisateur



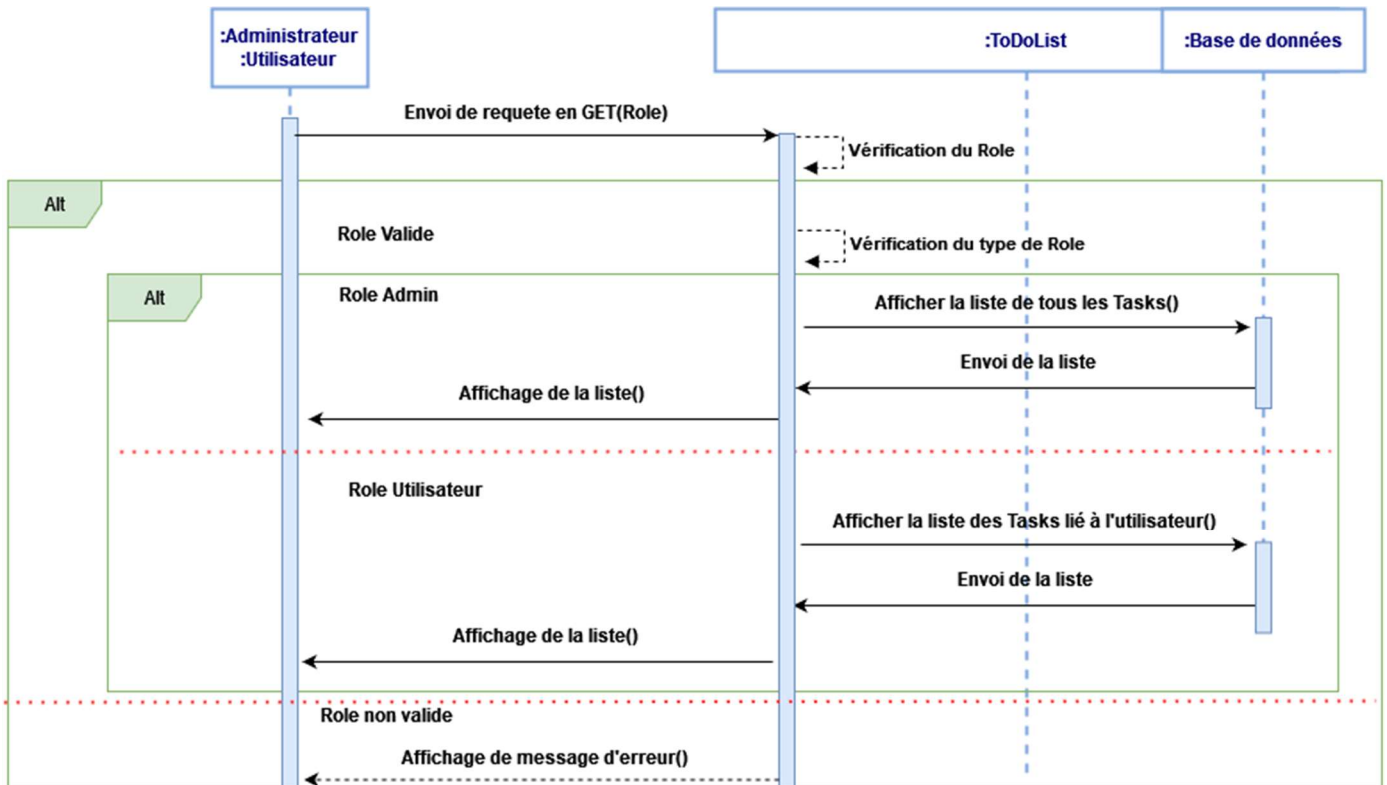
5.3 Mise à jour d'un utilisateur



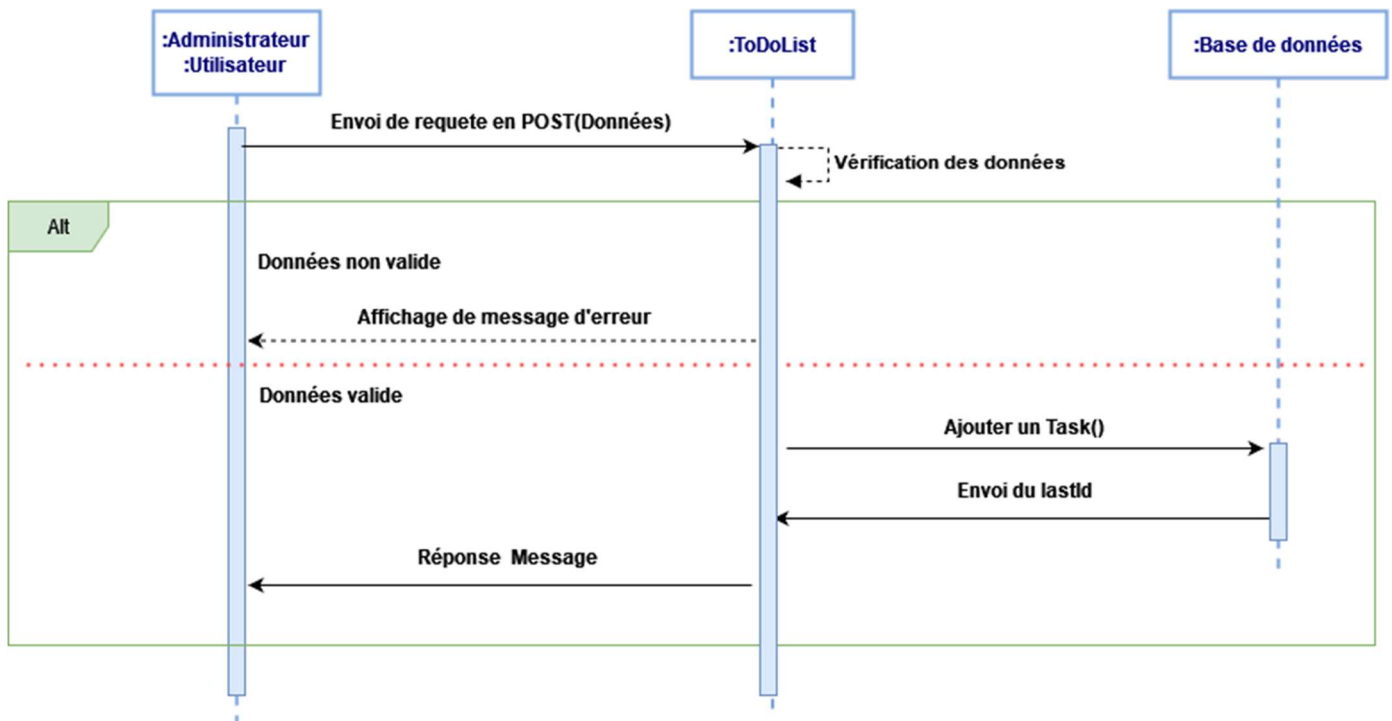
5.4 Lister des utilisateurs



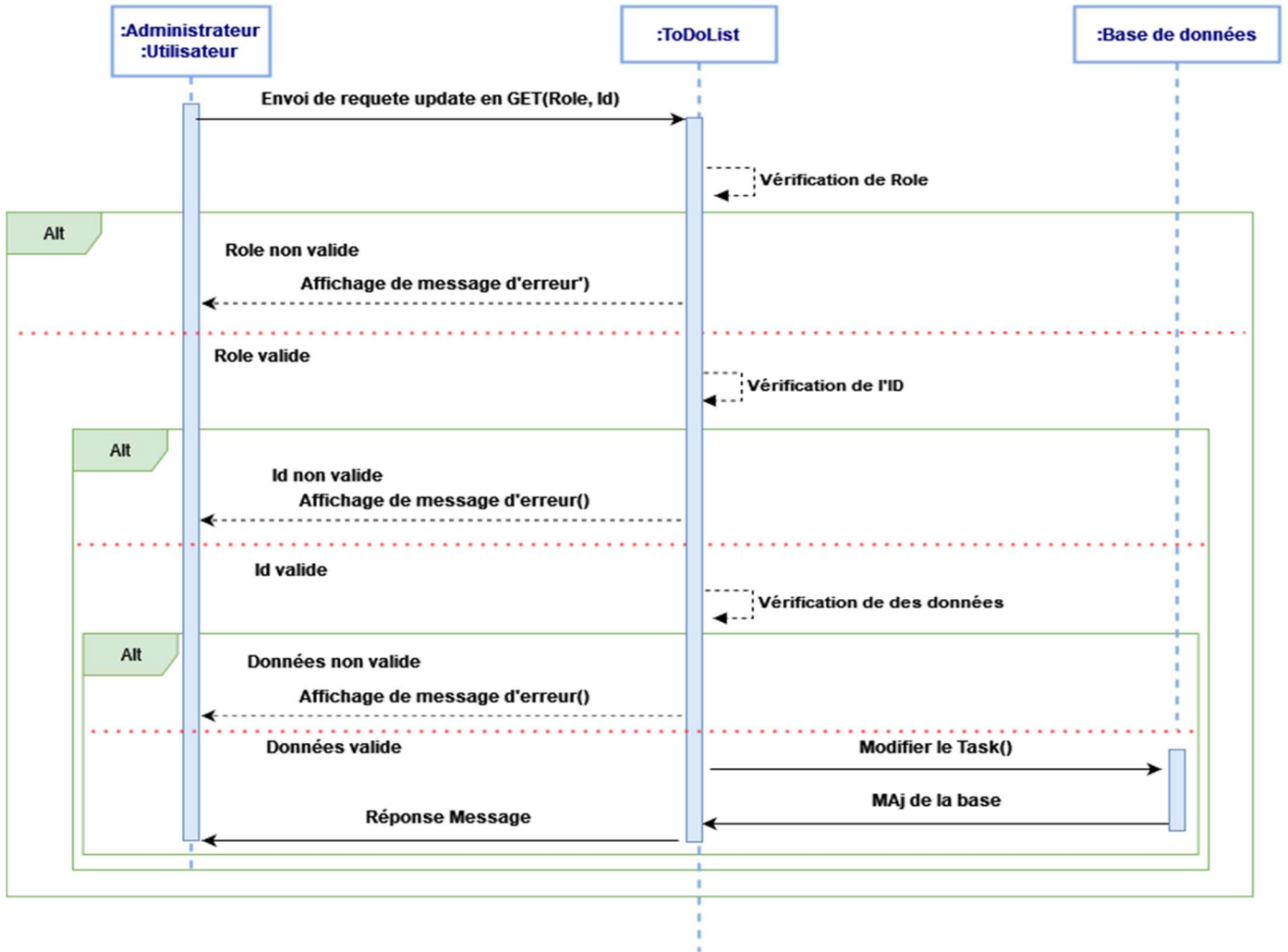
5.5 Lister les Tasks



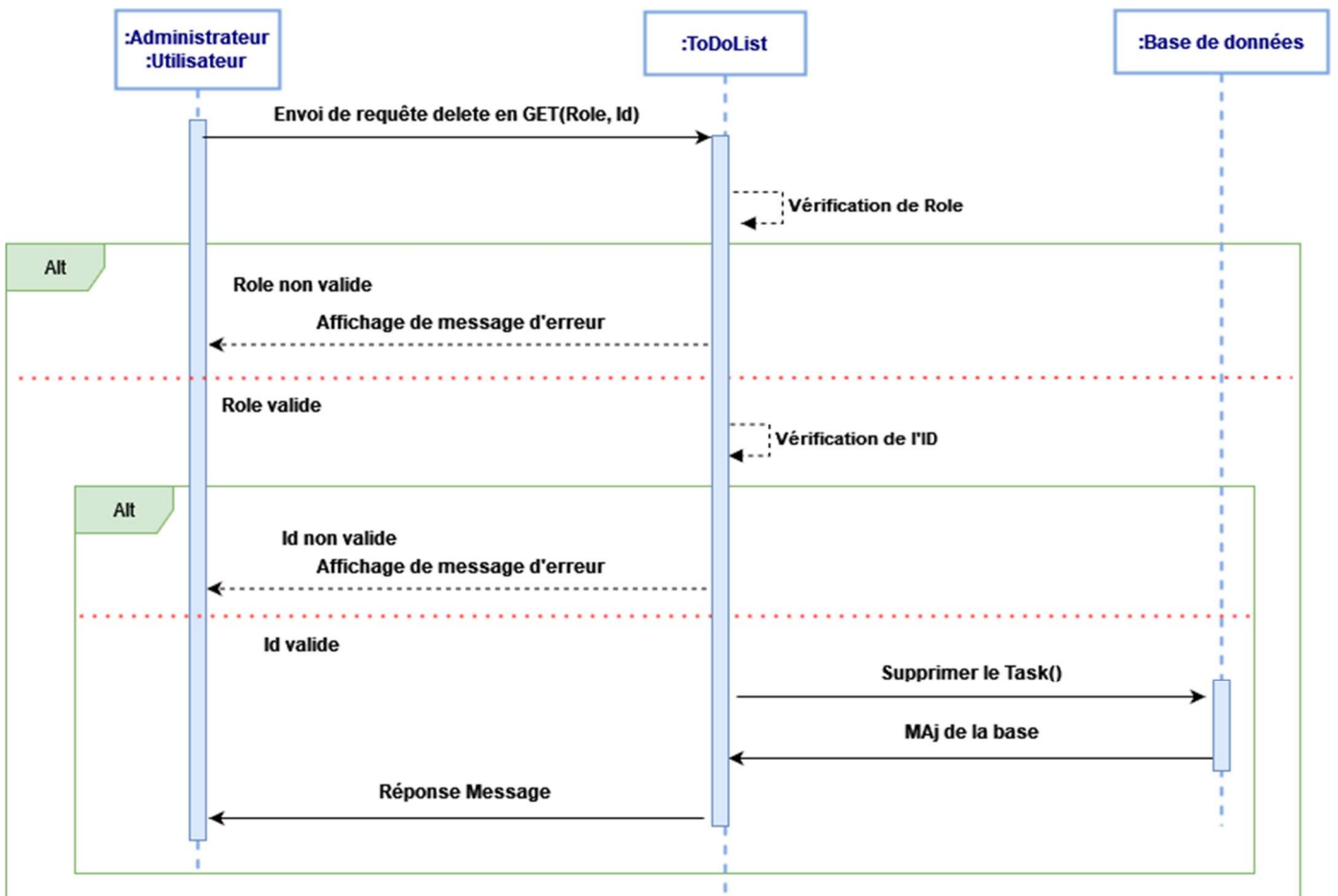
5.6 Ajout d'un Task



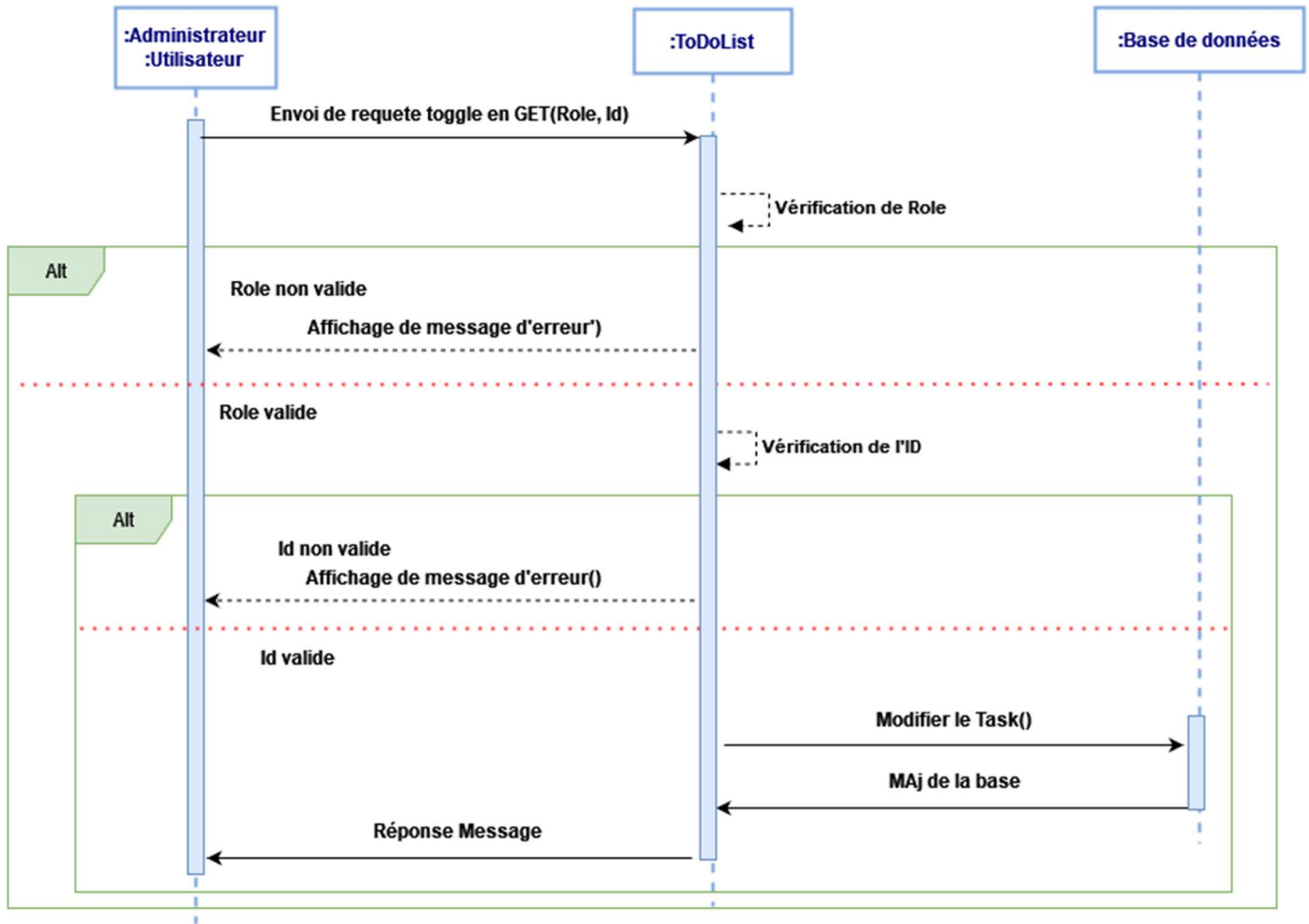
5.7 Modification d'un Task



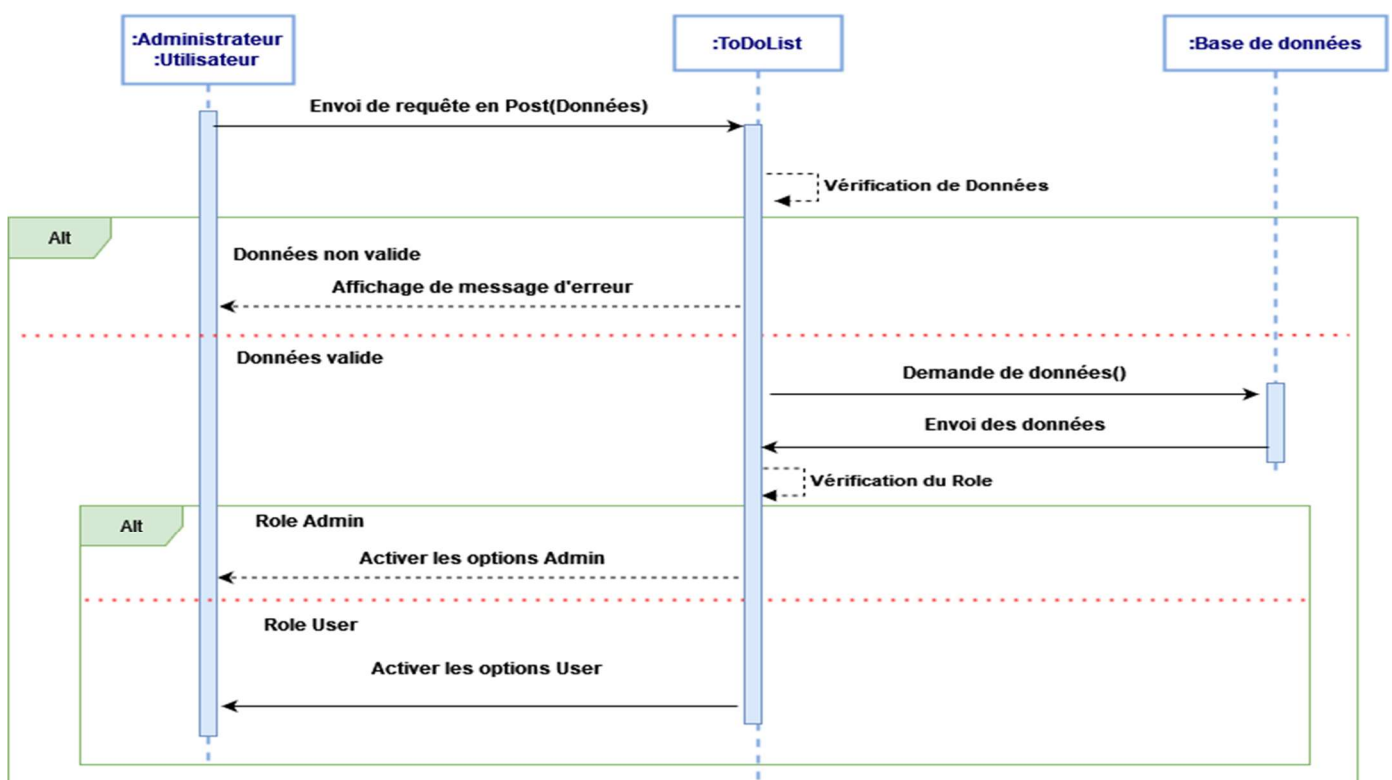
5.8 Suppression d'un Task



5.9 Modifier le statut d'un task



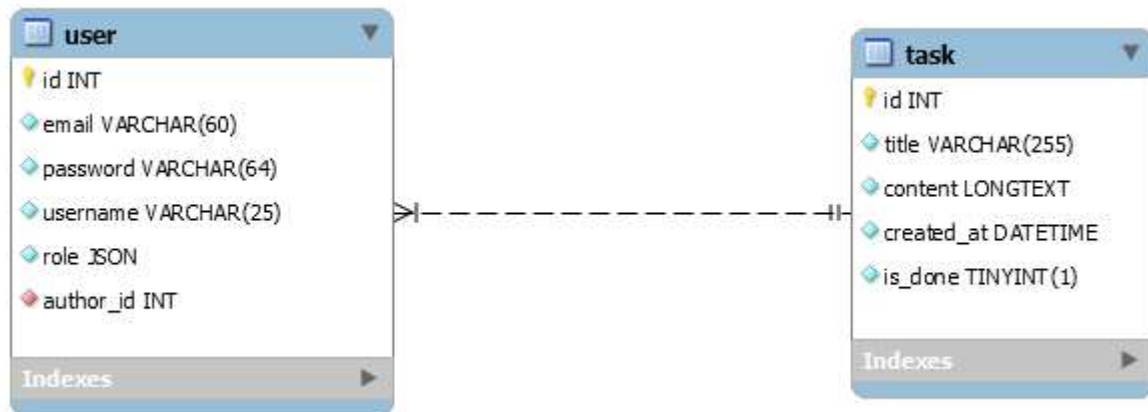
5.10 Authentification



6 Diagramme de classes



7 Modèle physique de données



8 Conclusion

Certaines zones d'ombres resteront à éclaircir, néanmoins ce document pourra être considéré comme un préambule tant aux demandes qu'aux éventuelles idées d'améliorations pouvant survenir

Un échange constructif est donc plus que demandé en vue d'améliorations.

De ce fait, j'ai modélisé une **base de données « MySQL » avec un jeu de données de démo.**

9 Proposition d'améliorations

Un ensemble d'améliorations est proposé pour ne citer que quelques-unes

1. Une double authentification
2. Un design plus attrayant
3. Une page pour lister le détail d'un **Task**
4. Mise en place des outils d'intégration continue.
5. Mise en place de « **dead line** » pour les **Tasls**

10 Annexe

- 1- <https://app.diagrams.net> à été utilisé dans le but de créer mes diagrammes de cas, les diagrammes de séquences ainsi que le diagramme de classe
- 2- **MySQL Workbench** a été utilisé dans le but de créer mon MPD « Model Physique de données »
- 3- Les tests automatisés sont implémentés avec **PHPUnit**
- 4- L'audit de performance est fait avec **Blackfire**
- 5- Le code qualité vérifié par « **Codacy**»
- 6- La vérification de l'impact climatique du code vérifié par « **CodeClimate** »