SoftDesk
Support

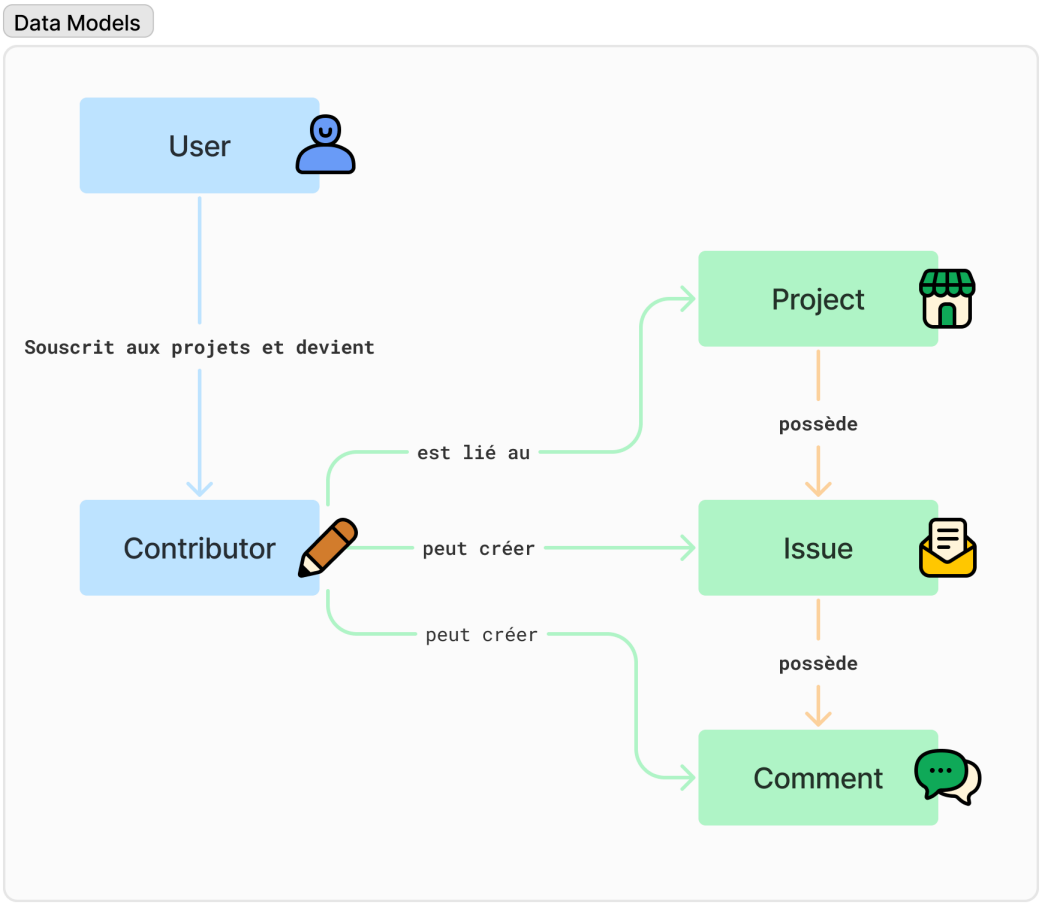
Conception de la mise en œuvre

Ce document fournit tous les détails nécessaires pour vous aider dans la réalisation de l'API RESTful pour SoftDesk Support.

Cette première section vous aidera à comprendre la logique métier de l'API REST que nous voulons développer, et notamment :

1. Un diagramme libre pour identifier les différentes ressources ;
2. Les définitions des ressources ;
3. Une liste des fonctionnalités requises pour l'application ;
4. Des recommandations pour tester les points de terminaison de l'application.

Diagramme des relations du système de suivi des problèmes



Définition des Ressources

User

Définit les utilisateurs, avec leur âge, leur choix de consentement et leurs identifiants...

Contributor

Définit les utilisateurs qui sont contributeurs d'un projet spécifique... Un utilisateur peut contribuer à plusieurs projets, et un projet peut avoir plusieurs contributeurs. Le contributeur peut créer trois types de ressources: le `project`, l'`issue` et le `comment`.

Project

Définit les projets d'une application cliente. C'est la ressource principale utilisée par le client.

Issue

Définit les problèmes d'un projet, ainsi que son statut, sa priorité, son attribution (utilisateur auquel le problème est affecté), sa balise (bug, tâche, amélioration). Un `project` peut posséder plusieurs `issues`, mais une `issue` n'est rattachée qu'à un seul `project`.

Comment

Définit les commentaires d'un problème (`issue`) particulier. Une `issue` peut avoir plusieurs `comments`, mais un `comment` n'est rattaché qu'à une seule `issue`.

Fonctionnalités de l'application

Gestion des utilisateurs

- L'application prend en compte les choix de confidentialité de chaque utilisateur en implémentant deux attributs : `can_be_contacted` (peut être contacté) : oui ou non, et `can_data_be_shared` (peut-on partager les données) : oui ou non.
- Selon les normes RGPD, un utilisateur doit avoir plus de 15 ans pour collecter ses données avec son consentement. Pour cela, nous récupérons et vérifions son âge lors de l'inscription.
- Un utilisateur peut s'authentifier à l'application grâce à un username et un password. L'authentification retourne un Json Web Token.
- L'utilisateur s'identifie dans chaque requête grâce au Json Web Token.

Gestion des projets

- Un utilisateur peut créer un projet. Il en devient l'auteur et le contributeur.
- Lors de la création, l'utilisateur doit pouvoir nommer le projet, ajouter une description ainsi qu'un type (*back-end*, front-end, *iOS* ou *Android*).
- Le contributor est une ressource spécifique, qui lie un utilisateur à un projet.

- Seuls les contributeurs d'un projet peuvent accéder à ce dernier. Seuls les contributeurs peuvent accéder aux ressources qui référencent un projet (l'issue et le comment).

Créations des tâches et des problèmes

- Un contributeur qui travaille sur un projet doit pouvoir créer des Issues (tâches/problèmes). Ces issues permettent de planifier des fonctionnalités à mettre en œuvre ou des bugs à régler dans un projet donné.
- Lors de la création de l'issue, le contributeur doit pouvoir la nommer et ajouter une description. Il doit aussi pouvoir assigner l'issue à un autre contributeur s'il le souhaite. Attention, seuls les contributeurs du projet correspondant à l'issue sont sélectionnables.
- Nous pouvons donner une priorité à l'issue (LOW, MEDIUM ou HIGH) pour connaître son importance.
- Nous pouvons aussi donner une balise (BUG, FEATURE ou TASK) pour connaître la nature de l'issue.
- Enfin, les contributeurs doivent pouvoir émettre un statut de progression (To Do, In Progress ou Finished). Par défaut, une issue est en To Do.

Créations des commentaires pour faciliter la communication

- Afin de mieux cerner les problèmes et faciliter la communication, les contributeurs d'un projet peuvent commenter les issues de ce projet grâce aux comments.
- L'auteur d'un commentaire écrit un texte qui sera sauvegardé en tant que description.
- Il doit aussi donner un lien vers une issue.
- Enfin, un identifiant unique de type uuid est automatiquement généré. Ce dernier permet de mieux référencer le comment.

Informations complémentaires

- Chaque ressource doit posséder un horodatage (created_time), afin de savoir quand la ressource a été créée.

Définition des auteurs

- Chaque ressource hors utilisateur doit posséder un author (auteur).
- L'auteur d'une ressource peut modifier ou supprimer cette ressource. Les autres utilisateurs ne peuvent que lire la ressource.

Mise en place de la pagination

- Un système de pagination est implémenté pour le listage des ressources.

Tester les points de terminaison d'API

Tous les points de terminaison peuvent être vérifiés à l'aide de l’application Postman (recommandée) ou de n'importe quel autre outil, comme la commande “curl” ou le serveur localhost du Django REST Framework.