Questions / Réponses sur ce projet

Questions sur la Conception et les Exigences

Quel est l'objectif principal de votre projet SoftDesk?

Réponse: L'objectif est de créer une API sécurisée RESTful pour remonter et suivre des problèmes techniques pour des entreprises en B2B.

Quelles sont les normes de sécurité que votre API doit respecter?

Réponse: Les normes OWASP et RGPD.

Quels sont les modèles principaux que vous avez implémentés?

Réponse: Les modèles Issue et Comment.

Quelles permissions spéciales sont requises pour les commentaires?

Réponse: Seul l'auteur d'un commentaire peut le mettre à jour ou le supprimer.

Quelle est la durée estimée pour compléter ce projet?

Réponse: 80 heures.

Quelle est la date de la dernière mise à jour du document d'exigences?

Réponse: Le document a été mis à jour le jeudi 20 juillet 2023.

Quelles sont les étapes clés pour tester votre application?

Réponse: Tester les différentes URL de l'application pour s'assurer de son bon fonctionnement.

Quelle est la durée recommandée pour la présentation de la soutenance?

Réponse: La présentation devrait durer 15 minutes (+/- 5 minutes).

Quelles sont les méthodes HTTP que votre API prend en charge pour les problèmes?

Réponse: GET pour récupérer, POST pour créer, PUT pour mettre à jour, et DELETE pour supprimer.

Quelles sont les méthodes HTTP que votre API prend en charge pour les commentaires?

Réponse: GET pour récupérer, POST pour créer, PUT pour mettre à jour, et DELETE pour supprimer.

Quels sont les champs du modèle Issue?

Réponse: title, desc, tag, priority, status, created_time.

Quels sont les champs du modèle Comment?

Réponse: description, created_time.

Comment avez-vous géré les relations entre les modèles?

Réponse: J'ai utilisé des clés étrangères pour relier les modèles entre eux.

Quelle bibliothèque avez-vous utilisée pour les sérialiseurs?

Réponse: Django REST Framework.

Comment avez-vous implémenté les permissions dans votre projet?

Réponse: J'ai créé un fichier permissions.py et défini des classes de permissions personnalisées.

Quelle stratégie de tests avez-vous adoptée?

Réponse: J'ai défini des chemins utilisateurs et testé les différentes URL de l'application.

Comment avez-vous optimisé les appels à votre API?

Réponse: J'ai minimisé les appels API grâce aux sérialiseurs.

Quelle est la structure de votre projet?

Réponse: Le projet est structuré en plusieurs dossiers et fichiers, y compris models.py, views.py, urls.py, etc.

Quelles sont les dépendances de votre projet?

Réponse: Les dépendances sont listées dans le fichier requirements.txt.

Comment avez-vous géré la documentation de votre API?

Réponse: Directement avec les options existantes dans POSTMAN.

Questions sur l'Implémentation et les Tests

Quel outil avez-vous utilisé pour tester votre API?

Réponse: J'ai utilisé Postman pour effectuer tous mes tests.

Comment avez-vous testé l'authentification dans Postman?

Réponse: J'ai utilisé des tokens JWT pour tester l'authentification.

Quels types de tests avez-vous effectués avec Postman?

Réponse: J'ai testé les opérations CRUD pour les modèles Issue et Comment.

Comment avez-vous testé les permissions dans Postman?

Réponse: J'ai utilisé différents utilisateurs pour tester les niveaux d'accès et les permissions.

Avez-vous rencontré des problèmes lors des tests avec Postman?

Réponse: (Cela dépend de votre expérience personnelle).

Comment avez-vous géré les erreurs dans votre API?

Réponse: J'ai utilisé des codes d'état HTTP pour indiquer le succès ou l'échec des requêtes.

Quelle méthode avez-vous utilisée pour filtrer les problèmes en fonction de l'ID du projet?

Réponse: J'ai utilisé la méthode filter sur le queryset pour filtrer les problèmes par project_id.

Comment avez-vous implémenté la création d'un nouveau problème dans votre API?

Réponse: J'ai utilisé la méthode create dans IssueViewset.

Comment avez-vous implémenté la suppression d'un commentaire?

Réponse: J'ai utilisé la méthode destroy dans CommentViewset.

Comment avez-vous récupéré un commentaire spécifique via son ID?

Réponse: J'ai utilisé la méthode retrieve dans CommentViewset.

Comment avez-vous mis à jour un commentaire existant?

Réponse: J'ai utilisé la méthode update dans CommentViewset.

Quelle est la réponse HTTP retournée après la suppression réussie d'un commentaire?

Réponse: HTTP 204 No Content.

Quels sont les choix possibles pour le champ tag dans le modèle Issue?

Réponse: (Cela dépend de votre implémentation, non mentionné dans les documents).

Quels sont les choix possibles pour le champ priority dans le modèle Issue?

Réponse: (Cela dépend de votre implémentation, non mentionné dans les documents).

Quels sont les choix possibles pour le champ status dans le modèle Issue?

Réponse: (Cela dépend de votre implémentation, non mentionné dans les documents).

Comment avez-vous géré les relations de clé étrangère dans le modèle Issue?

Réponse: J'ai utilisé des clés étrangères pour relier Issue aux modèles Project, User (pour l'auteur), et User (pour l'assigné).

Quelle est la longueur maximale du champ description dans le modèle Comment?

Réponse: La longueur maximale est de 255 caractères.

Comment le champ created_time est-il géré dans le modèle Comment?

Réponse: Il est automatiquement défini au moment de la création du commentaire.

Quelle est la structure de votre projet en termes de dossiers et de fichiers?

Réponse: Le projet a une structure de dossiers comprenant softdesk, API_SOFTDESK, et des fichiers comme models.py, views.py, urls.py, etc.

Quel fichier contient la liste de toutes les dépendances de votre projet?

Réponse: Le fichier requirements.txt.

Questions sur la Sécurité et la Documentation

Comment avez-vous implémenté l'authentification dans votre API?

Réponse: J'ai utilisé JWT (JSON Web Tokens) pour l'authentification.

Quelle bibliothèque avez-vous utilisée pour JWT?

Réponse: J'ai utilisé djangorestframework-simplejwt.

Comment avez-vous géré les autorisations dans votre API?

Réponse: J'ai créé des classes de permissions personnalisées dans permissions.py.

Quelle méthode avez-vous utilisée pour sécuriser les données sensibles?

Réponse: J'ai utilisé HTTPS et des variables d'environnement pour sécuriser les données sensibles.

Avez-vous utilisé des tests automatisés pour la sécurité?

Réponse: Oui, tout est dans POSTMAN.

Comment avez-vous géré la pagination dans votre API?

Réponse: J'ai utilisé les fonctionnalités de pagination de Django REST Framework.

Avez-vous implémenté des taux limites pour votre API?

Réponse: (Non, cela est prévue dans les améliorations futures).

Comment avez-vous géré la validation des données entrantes?

Réponse: J'ai utilisé des sérialiseurs pour la validation des données.

Comment avez-vous géré les erreurs de validation?

Réponse: J'ai renvoyé des réponses HTTP avec des messages d'erreur appropriés.

Avez-vous documenté votre API?

Réponse: Oui, en totalité avec POSTMAN.

Si oui, quelle méthode avez-vous utilisée pour la documentation?

Réponse: (Automatique depuis POSTMAN).

Avez-vous utilisé des variables d'environnement pour stocker des informations sensibles?

Réponse: Oui, j'ai utilisé des variables d'environnement pour stocker des clés secrètes et des informations de base de données.

Comment avez-vous géré les dépendances de votre projet?

Réponse: J'ai utilisé un fichier requirements.txt pour gérer les dépendances.

Avez-vous utilisé des hooks Git pour automatiser des tâches?

Réponse: Oui, bien évidemment, et ce depuis le debut.

Comment avez-vous géré les migrations de la base de données?

Réponse: J'ai utilisé les commandes makemigrations et migrate de Django.