Guide d'étapes clés : Créez une API sécurisée RESTful en utilisant Django REST

Comment utiliser ce document ?

Ce guide vous propose un découpage du projet en étapes. Vous pouvez suivre ces étapes selon vos besoins. Dans chacune, vous trouverez :

- des recommandations pour compléter la mission ;
- les points de vigilance à garder en tête ;
- une estimation de votre avancement sur l'ensemble du projet (attention, celui-ci peut varier d'un apprenant à l'autre).

Suivre ce guide vous permettra ainsi :

- d'organiser votre temps ;
- de gagner en autonomie ;
- d'utiliser les cours et ressources de façon efficace ;
- de mobiliser une méthodologie professionnelle que vous pourrez réutiliser.

Gardez en tête que votre progression sur les étapes n'est qu'une estimation, et sera différente selon votre vitesse de progression.

Recommandations générales

- Prenez le temps de bien cadrer le projet avant de le démarrer. Ne le commencez pas s'il comporte encore trop de zones d'ombres. Bien entendu, vous pourrez en éclaircir certaines lors de l'implémentation, mais une bonne préparation vous permettra d'avancer dans de meilleures conditions.
- Django Rest Framework est une surcouche à Django et permet entre autres d'automatiser la gestion CRUD (Create, Read, Update, Delete) de vos modèles sous forme d'API. Prenez le temps d'assimiler ses mécanismes et utilisez toute sa puissance en implémentant les ModelViewsets.
- Déterminez une stratégie de tests en définissant des chemins utilisateurs. Une API REST ne retourne que de la donnée brute, mais vous pouvez tout de même savoir quelle requête sera utile par quel utilisateur. Par exemple, la création de

l'utilisateur est propre à la phase d'inscription. La création d'un contributeur est propre à la souscription d'un utilisateur vers un projet. La création d'un commentaire se fait lorsqu'un contributeur souhaite réagir à un problème particulier... Réfléchir de cette manière permettra de donner du sens à votre implémentation et à vos tests.

Étape 1 : Démarrez le projet et identifiez le besoin

10 % de progression

Avant de démarrer cette étape, je dois avoir :

- étudié mes notes ainsi que les documents Conception de la mise en oeuvre et Exigences de sécurité et d'optimisation ;
- identifié les modèles d'objets dans le diagramme des relations du système de suivi des problèmes du document Conception de la mise en œuvre.

Une fois cette étape terminée, je devrais avoir :

- configuré le projet ;
- versionné le projet sur Github, avec un fichier README expliquant l'installation et le lancement du projet en local.

Recommandations :

- Prenez le temps de bien comprendre le besoin ;
- Le diagramme libre dans le document Conception de la mise en œuvre, sans formalisme particulier, permet d'identifier les différents modèles de données présents dans l'application. C'est sur cette base que vous pourrez, si vous le souhaitez, élaborer des diagrammes plus précis, de classe ou d'entité/relation, par exemple. Dans tous les cas, il peut être intéressant de se renseigner sur les différents types de diagrammes qui peuvent aider à la conception d'une application ;
- Ces différentes ressources se caractérisent par des modèles dans l'ORM de Django, et définissent à la fois la table SQL et la classe Python. Il sera nécessaire de penser les différents attributs pour chaque modèle ;
- Le diagramme propose aussi des liens entre chaque ressource, qui caractérisent les différentes relations qu'elles peuvent avoir entre elles. Encore une fois, ce sera à vous de définir la nature de ces relations (*One To One*, *One To Many* ou *Many To Many*...) ;
- Créez les attributs nécessaires à chacun de vos modèles ;
- Remplacez votre utilisation de pip par Pipenv ou Poetry afin de sécuriser la gestion des dépendances entre elles.

Points de vigilance :

- **Attention** : une erreur serait de se plonger directement dans le code alors que le projet n'a pas été assez cadré.

Ressources :

- La mise en place d'un projet Django et Git et Github de la documentation Django ;
- Le cours Débutez avec le framework Django ;
- Comment configurer son projet Django de la documentation Django ;
- La documentation de Django Rest Framework (rédigée en anglais) ;
- Le cours Modélisez vos bases de données ;
- La documentation sur Github Dependabot (rédigée en anglais).

Étape 2 : Définissez les utilisateurs

20 % de progression

Avant de démarrer cette étape, je dois avoir :

- défini les attributs des modèles de données ;
- pensé et spécifié les différentes routes que les utilisateurs pourront prendre.

Une fois cette étape terminée, je devrais avoir :

- mis en place le modèle User.

Recommandations :

- Pensez à créer une app Django spécifique pour le modèle User ;
- Redéfinissez le modèle User de base de Django. C'est une bonne pratique encouragée par l'équipe de Django ;
- Créez le modèle, puis serialisez-le. Passez enfin à la vue (renseignez-vous le système de ModelViewset de Django Rest Framework) ;
- Finalisez l'implémentation avec la mise en place du router ;
- Vérifiez l'implémentation par le biais de l'application Postman, et assurez-vous de pouvoir créer, lire, modifier et supprimer un utilisateur.

Points de vigilance :

- Attention à bien réfléchir aux relations entre les modèles (OneToMany, ManyToMany, OneToOne...). Ici, nous n'implémentons que le modèle User, mais lorsque vous implémenterez les autres modèles, il sera nécessaire de repenser les relations avec les modèles existants ;
- Il est important de vérifier le bon fonctionnement de chaque partie de son code. N'hésitez pas à utiliser la commande "python manage.py shell" pour entrer dans l'application Django et manipuler les objets, comprendre leurs intéractions ;

- Enfin, Postman est l'étape décisive pour vérifier que le serveur répond aux requêtes utilisateurs ;
- Attention, dans le respect des normes RGPD, un utilisateur de moins de 15 ans ne devrait pas pouvoir finaliser son inscription.

Ressources :
- La documentation sur les applications de Django ;
- La première partie du cours Mettez en place une API avec Django REST Framework ;
- La documentation sur les routers de Django REST Framework (rédigée en anglais) ;
- La documentation de Postman (rédigée en anglais).

Étape 3 : Définissez les projets et les contributeurs

40 % de progression

Avant de démarrer cette étape, je dois avoir :
- implémenté et testé le modèle User.

Une fois cette étape terminée, je devrais avoir :
- implémenté et testé les modèles Application et Contributor.

Recommandations :
- Réfléchissez au découpage de votre projet. Devez-vous créer une ou plusieurs app Django ? Le contributeur est-il plus proche du modèle utilisateur ou du modèle projet ? Un bon découpage du code permet de mieux le comprendre et ainsi, de faciliter sa mise en place ;
- Implémentez le modèle Project avant le modèle Contributor. En effet, le Contributor ne peut fonctionner sans Project ;
- Pensez au cas où le Contributor serait aussi l'auteur du projet (son créateur).

Point de vigilance :
- Encore une fois, prenez le temps de bien tester les différentes URL de votre application pour vous assurer de son bon fonctionnement.

Étape 4 : Définissez les problèmes et les commentaires

60 % de progression

Avant de démarrer cette étape, je dois avoir :
- implémenté et testé les modèles Project et Contributor.

Une fois cette étape terminée, je devrais avoir :

- implémenté et testé les modèles Issue et Comment.

Recommandation :

- L'implémentation de ces deux modèles de données est relativement similaire. La mise en place de l'un facilite donc celle de l'autre.

Points de vigilance :

- Pensez à l'architecture de votre code ;
- Testez chaque point de terminaison (URL de l'application).

Étape 5 : Mettez en place le système de permissions

75 % de progression

Avant de démarrer cette étape, je dois avoir :

- implémenté et testé les différents modèles de données de l'application.

Une fois cette étape terminée, je devrais avoir :

- implémenté et testé le système de permissions ;
- ajouté le dependabot au repository Github.

Recommandations :

- Il est maintenant temps d'ajouter des permissions à l'application ! Commencez par ajouter la permission d'authentification à l'aide d'un Json Web Token ;
- Implémentez ensuite les permissions de lecture et d'écriture sur chaque ressource. L'auteur d'une ressource a tous les droits dessus, mais les autres utilisateurs ne peuvent que lire la ressource ou la référencer dans une autre ressource ;
- Utilisez le système de classes de permissions fourni dans Django Rest Framework pour créer des permissions sur les vues ;
- Pensez aussi à la confidentialité des données : vérifiez bien que l'utilisateur possède les champs définis dans notre application des normes RGPD.

Points de vigilance :

- Il ne s'agira plus de tester simplement les opérations Create, Read, Update, Delete sur chaque ressource, mais de vérifier ces différentes opérations selon le type d'utilisateur (non authentifié, authentifié, contributeur, auteur...) ;
- Le « droit à l'oubli » est une règle qui dit qu'un utilisateur doit pouvoir supprimer ses données personnelles sans subsistance dans la base de données de l'application cible. Dans le cadre de cette application, ce droit à l'oubli se résout automatiquement lors de la suppression, mais certaines

applications remplacent la suppression réelle par une « fausse suppression », appelée « soft delete ».

Ressources :

- Le guide pratique RGPD de la CNIL ;
- L'article Le droit à l'effacement : supprimer vos données en ligne de la CNIL ;
- Le chapitre Qu'est-ce que le RGPD ? du cours Maîtrisez les risques juridiques du marketing et de la communication ;
- La partie 3 Sécurisez votre API avec l'authentification du cours Mettez en place une API avec Django REST Framework ;
- La documentation sur les permissions de Django REST Framework (rédigée en anglais).

Étape 6 : Pensez « green code » et optimisez l'application

90 % de progression

Avant de démarrer cette étape, je dois avoir :

- implémenté et testé les différents modèles de données de l'application ;
- sécurisé l'application.

Une fois cette étape terminée, je devrais avoir :

- optimisé l'application pour la rendre moins gourmande en ressources ;
- implémenté la pagination.

Recommandations :

- Vérifiez que votre application ne possède pas de trop grandes imbrications des ressources. Ce projet ne demande pas d'imbriquer les ressources entre elles, c'est donc à vous de décider si vous souhaitez ou non imbriquer les ressources dans le corps de réponse de la requête. Cependant, contentez-vous d'un seul niveau d'imbrication pour éviter de retourner des requêtes trop volumineuses ;
- Django Rest Framework possède un système de pagination puissant et simple à mettre en place. N'hésitez pas à parcourir la documentation associée.

Point de vigilance :

- Réfléchissez bien lors de l'implémentation de la pagination. Combien de ressources souhaitez-vous fournir par page ? Des pages trop petites demanderont de créer plus de requêtes pour accéder aux différentes ressources. Mais des pages trop grandes augmentent drastiquement le poids de chaque requête.

Ressources :

- Le chapitre Minimisez les appels à votre API grâce aux serialiseurs du cours Mettez en place une API avec Django REST Framework ;
- Le chapitre Réduisez l'empreinte écologique de votre site web du cours Appliquez les principes du green IT dans votre entreprise.

Projet terminé !