Simple blog with GraphQL API and VueJs

Proposition de sujet cfa des sciences - sujet de fin d'année

Introduction

Description générale

Pour le project tutorés de fin d'année, je propose la réalisation d'un blog assez simple sur le plan fonctionnel. Rédaction de *Post* par les *User*, les *Post* pourront être commentés (*Comment*), et on pourra répondre aux commentaires (*Reply*).

Le site web sera composée d'une page principale, listant tous les articles, et une page "profile", listant les informations du *User* connecté.

Le projet aura principalement un enjeu technique, avec la découverte de nouvelles technologies et la mise en place d'un environnement automatisé et complet, du développement jusqu'à la mise en production.

Enjeux - technologies

L'un des enjeux du projet sera de séparer le *font-end* du *back-end*:

- Back-end : API GraphQL en Python/Django
- Front-end : VueJs

Pour ce projet nous souhaiterons aussi nous intéresser à l'automatisation du déploiement ainsi qu'à l'infrastructure, afin d'avoir une composante DevOps pour ce projet.

Pour ce faire nous utiliserons les technologies suivantes:

- Docker: conteneurisation pour faciliter la mise en place de l'environnement de déploiement ainsi que la séparation entre l'API, le front-end et la database.
- Capistrano: automatisation du déploiement, scripts, et lancement des conteneurs docker sur le serveur.
- Digital Ocean: service externe d'hébergement cloud.

Specifications

Classes

```
User {
  lastname
  firstname
  username
  email
  date_of_birth
  profil_pic
  #posts (One_to_Many > Post)
  ...
}
```

```
Post {
  title
  date
  body
  draft (boolean)
  #author (Many_to_One > User)
  #likes (Many_to_Many > User)
}

Comment {
  date
  body
  #author (Many_to_One > User)
  #related_post (Many_to_One > Post)
}

Reply {
  date
  body
  #author (Many_to_One > User)
  #related_comment (Many_to_One > Comment)
}
```

Ecrans

- Page d'accueil
  - Liste des articles (*Post*)
  - Actions user lambda: consulter(>page de l'article), commenter, enregistrer
  - Actions auteur: idem + éditer
- Page d'un article:
  - Actions user lambda: commenter, répondre à un commentaire, enregistrer
  - Actions auteur: idem + éditer
- Page nouvel article / édition article
  - Editeur de texte
  - Poster, enregistrer dans les brouillons, annuler
- Page profil: privée et publique
  - Consulter et éditer son profil
  - Consulter le profil des autres users
  - Liste des articles rédigés

Point technique

Ops

L'un des enjeux principal du projet sera aussi d'automatiser le déploiement du site. Les technologies qui seront utilisées ont été listé plus haut. Capistrano nous permettra de déployer très simplement en production notre site et d'avoir une trace des différentes versions (si besoin nous pourr ns revenir à une version précédente). Nous lancerons par ce biais les conteneurs docker, avec *docker-compose* qui possèderont chacun les environnements nécessaires à chaque partie de l'architecture.

C'est avec *docker-compose* que nous assurerons le lien entre l'API, le front-end et la database.

GraphQL

GraphQL est une technologie qui permet de développer des APIs. Le modèle a été inventé par Facebook et permet de réaliser des queries de manière plus efficace qu'avec une API REST. L'idée principale de ce projet est donc de prendre en main GraphQL à travers son implémentation en Python/Django.

Licence