**Introduction :**

Les moteurs de recherche portent essentiellement sur des filtres pour la richesse des informations disponibles sur Internet. Ils permettent aux utilisateurs de trouver rapidement et facilement des informations d'intérêt ou de valeur réelles, sans avoir à parcourir de nombreuses pages Web non applicables. Il y a beaucoup de filtrage à faire - il y a trois ans, en 2004, le nombre de pages dans l'index de Google a dépassé le nombre de personnes sur la planète, atteignant un chiffre stupéfiant de plus de 8 milliards. Avec autant de contenu, Internet serait essentiellement impraticable sans moteurs de recherche, et les internautes se noieraient dans une mer d'informations non applicables et de messages marketing stridents.

Les moteurs de recherche fournissent aux utilisateurs des résultats de recherche qui conduisent à des informations pertinentes sur des sites Web de haute qualité. Le mot clé ici est « pertinent ». Pour atteindre et conserver des parts de marché dans les recherches en ligne, les moteurs de recherche doivent s'assurer qu'ils fournissent des résultats pertinents pour les recherches de leurs utilisateurs. Ils le font en maintenant des bases de données de pages Web, qu'ils développent en utilisant des programmes automatisés connus sous le nom de « araignées » ou « robots » pour collecter des informations. Les moteurs de recherche utilisent des algorithmes complexes pour évaluer les sites Web et les pages Web et leur attribuent un classement pour les expressions de recherche pertinentes. Ces algorithmes sont jalousement gardés et fréquemment mis à jour.

Dans ce projet, il nous est donc demandé de réaliser un moteur de recherche sur la base de livres de Gutenberg, étant un site contenant une grosse quantité de livres diffèrent. Ainsi, il s’agira d’utiliser des technologies et des algorithmes afin d’obtenir un moteur de recherche efficace en proposant des résultats pertinent (en proposant des résultats adaptés à chaque recherche) sur une grosse quantité de données.

**Problématique :**

Une recherche directe sur une base de données volumineuses nécessite un temps considérable vu qu'on doit parcourir tous les livres un par un et dans chaque livre mot par mot, pour s'applique à ce problème, on doit utiliser le principe de fonctionnement des moteurs de recherche :

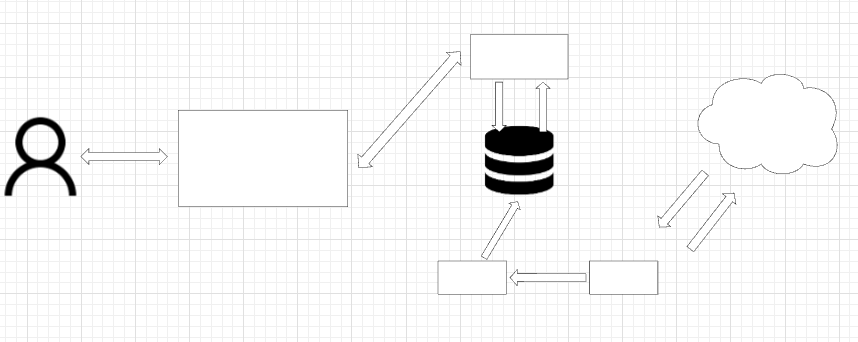
— Une indexation du contenu livres dans une base de données, est effectuée préalablement à la

recherché.

— Ainsi, le moteur restitue, en fonction de ses paramètres une liste de pages correspondant aux

mots recherchés.

Architecture :



L’architecture utilisé dans la figure 3 est appelée RESTful est un style d’architecture permettant

de construire des applications Web. Il s’agit de séparer le front et le backend dans deux composants

différents, et les faire communiquer via des requêtes HTTP à travers des API développée au niveau

du backend selon les services demandés.

De plus, on a utilisé l’api du site https://gutendex.com/pour récupérer l’ensemble des livres au

moment de l’indexation.

Technologie utilisée :

*Python Django*

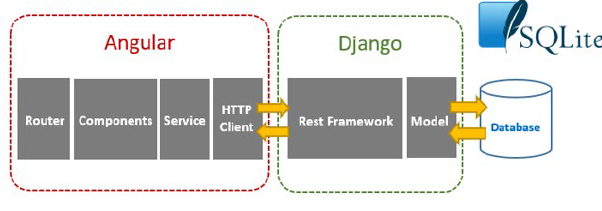
En ce qui concerne le développement backend , Python fait partie des choix les plus populaires que font les entreprises, ainsi que les développeurs. Récemment, de plus en plus de personnes sont passées à Python. Après sa mise en oeuvre pour les plates-formes de produits par certains des plus grands noms comme Instagram, YouTube, Quora, Pinterest, Facebook, Reddit, Netflix, Google et Spotify, Python est devenu le nouveau favori.

*2- Angular*

Devant une concurrence framework aussi intense, le besoin d'applications Web conviviales et interactives devient très crucial pour favoriser le succès de l'entreprise. Angular a servi de cadre idéal pour diriger l'industrie pour la production de solutions évolutives qui répondent aux besoins des gens. Angular est connu comme un framework Javascript open-source qui a été développé pour la première fois en 2009 par le développeur Google Misko Hevery. La structure basée sur CSS, HTML et JS afin de faciliter le développement front-end.

Les applications Web Angular JS ont connu un tel succès qu'environ 8 400 sites Web prennent en charge ce cadre. Certaines de ces sociétés incluent également Upwork, PayPal et Netflix.

Angular possède de nombreuses fonctionnalités qui permettent aux professionnels du développement d'applications Web de concevoir des sites Web avec un codage minimal. L'obligation d'écrire des setters et des getters dans différents modèles de données est supprimée. Moins de codage permet d'économiser beaucoup de temps et d'efforts.



- Django exporte les Apis REST à l'aide de Django Rest Framework et interagit avec la base de données SQLite à l'aide du modèle Django.

- Angular envoie des requêtes HTTP et récupère les réponses HTTP , affiche des données sur les composants.

BACKEND :

La partie backend représente le noyau du projet, cette partie implémente les différents algorithmes

qu’on verra ci-dessous qui permettent d’implémenter le moteur de recherche proposer aux

utilisateurs.

afin d’élaborer un bon moteur de recherche plusieurs étapes sont nécessaires :

1. Traitement des données & définition de la base de donnée utilisée

2. Indexation des données

3. Implémentassions des algorithmes de recherche qui utilise l’indexation.

4. Réalisation d’une API qui permet de transmettre au Frontend les résultats du moteur de

recherche.

Les applications Web Django accèdent et gèrent les données via des objets Python appelés modèles. Les modèles définissent la *structure* des données stockées, y compris les *types de* champs et éventuellement leur taille maximale, les valeurs par défaut, les options de liste de sélection, le texte d'aide pour la documentation, le texte d'étiquette pour les formulaires, etc. La définition du modèle est indépendante de la base de données sous-jacente, il est possible d’en choisir un parmi plusieurs dans le cadre des paramètres du projet.

Une fois avoir choisi la base de données à utiliser pour le projet, il suffira d’implémenter la structure du modèle et Django s'occupera de la communication avec la base de données.

Parler ici de la constructionn de la base, combien de livre quel données etc

Structure de donnée

1. **BookModel** : qui définit une donnée indexer avec :

— id : l’identifiant du livre indexé

— author : les différents auteurs du livre

— language : la langue de saisit du livre

— title : le titre du livre de type

— CoverBook : représente la photo de couverture du livre

— text : le contenu du livre

— crank : représente le score du livre qui serra utilisé lors de l’algorithme de classement.

2. **BookIndexModel** : qui représente le résultat de l’algorithme décrit dans la sections cidessous

les données sont stockés comme suit :

— id : identifiant d’une indexation sur un mot.

— word : le mot indexer sur l’ensemble des livres contenu dans la base de donnée.

— idBook : id du livre ou le mot figure.

— occurence : nombre d’occurrence du mot **word** dans le livre avec un **idBook**

Après avoir défini nos modèles => indexage

**4.2 algorithme d’indexation**

et créé quelques enregistrements de bibliothèque initiaux avec lesquels travailler, il est temps d'écrire le code qui présente ces informations aux utilisateurs. La première chose à faire est de déterminer les informations que nous voulons afficher dans nos pages et de définir les URL à utiliser pour renvoyer ces ressources. Ensuite, nous créerons un mappeur d'URL, des vues et des modèles pour afficher les pages.

Algorithme utilisée

Crank, jaccard etc..

Modele

Front End