

Enseignant cadrant : Yassine BOUFENNECHE

Promo : 2022-2023

Classe : A3MSI



CAHIER DES CHARGES

Plateforme de génération de l'état de l'art dans
les articles et documents scientifiques

+ DERNONCOURT Inès
+ KASDI Anthénéa
+ MUTHUKRISHNAN Cindy
+ HOSSAIN Ishika

Table des matières

Introduction.....	3
I. Description du projet.....	4
A. Qu'est-ce que l'état de l'art ?	4
B. Objectifs du projet	4
II. Exigences fonctionnelles	5
A. Liste des fonctionnalités	5
B. Liste des fonctionnalités optionnelles.....	5
III. Organisation de l'application Web.....	6
IV. Vue technique du projet.....	6
A. Front-end et Back-end.....	6
B. Outils Utilisés	7
C. Base de données	7
D. Design de l'API	9
E. Angular.....	10
F. Spring Boot	12

Introduction

Dans le cadre du projet de fin d'étude de l'École d'Ingénieurs ESME Sudria, nous avons fait le choix de traiter le sujet concernant l'automatisation de l'état de l'art dans les articles et documents scientifiques

La partie "état de l'art" est une partie inévitable lorsque l'on souhaite écrire un article scientifique. Elle correspond à ce qui a été fait jusqu'à présent sur un sujet spécifique et permet de poser une base de réflexion pour l'écriture de l'article.

Autrement dit, elle permet de savoir où en est l'Homme dans ses connaissances sur un domaine spécifique. La rédaction de l'état de l'art peut donc s'avérer être une étape longue et complexe.

Le but de notre projet est donc de créer une plateforme qui permet de générer un état de l'art pour un domaine particulier en fonction de mots clés. Cela facilitera les utilisateurs ou les chercheurs scientifiques dans la rédaction de leurs futurs articles.

Ce rapport présentera le projet dans sa globalité en formalisant le besoin et en définissant tous les aspects du projet (fonctionnel, organisationnel, technique, etc.).

I. Description du projet

A. Qu'est-ce que l'état de l'art ?

Afin de réaliser ce projet, il est d'abord primordial de comprendre ce que l'état de l'art.

Dans les écrits scientifiques, l'état de l'art décrit les connaissances actuelles sur le sujet étudié à travers l'analyse de travaux similaires ou connexes publiés. Il peut fournir une vue d'ensemble de ce qui a été fait dans le domaine et de ce qui devrait être étudié plus en profondeur, afin d'aider à formuler les problèmes et les hypothèses que l'article entend aborder.

Aujourd'hui, nombreuses sont les personnes essayant de réaliser un bon état de l'art, qui est considéré comme la principale étape initiale d'une thèse de doctorat par exemple. Toutefois, il s'agit d'une tâche complexe qui implique l'analyse, la comparaison, l'évaluation et la mise en relation de différentes sources, c'est-à-dire, de nombreuses heures de lecture et d'organisation du contenu.

Cette tâche nécessite également d'être rafraîchi de manière continue et incrémentale car si commencer un article en produisant un bon état de l'art est un bon point de départ, ce processus n'est pas linéaire et implique de nombreuses itérations.

C'est pourquoi notre plateforme a pour but de faciliter ce processus long et complexe.

B. Objectifs du projet

Les objectifs principaux du projet sont donc de :

- Développer une plateforme qui permet de générer un état de l'art pour un domaine particulier en fonction de mots clés.
- Permettre aux auteurs de « résumer » une grande partie de leur travail, qui sera exploité par d'autres chercheurs.

En plus de générer automatiquement l'état de l'art, la plateforme sera aussi un meilleur moyen pour les chercheurs de se positionner par rapport à un sujet donné avec moins d'effort.

II. Exigences fonctionnelles

A. Liste des fonctionnalités

L'idée principale étant de créer une application web permettant aux internautes de :

- S'inscrire sur le site en fournissant un nom d'utilisateur, un email, un mot de passe, sa date de naissance et d'autres informations complémentaires.
- Se connecter sur le site en tant qu'auteur ou chercheur, selon le but de l'utilisateur.
- Déposer des articles en tant qu'auteur.
- Consulter des articles pour en générer l'état de l'art en tant que chercheur.
- Consulter les articles en relation avec les mots-clés tapés dans la barre de recherche.
- Stocker les informations sur les auteurs : nom, prénom, université/faculté, ...
- Stocker chaque article, résumé du site.
- Contacter notre équipe en cas de besoin via le formulaire de contact.
- Prendre connaissance des questions les plus posées via la page FAQ.

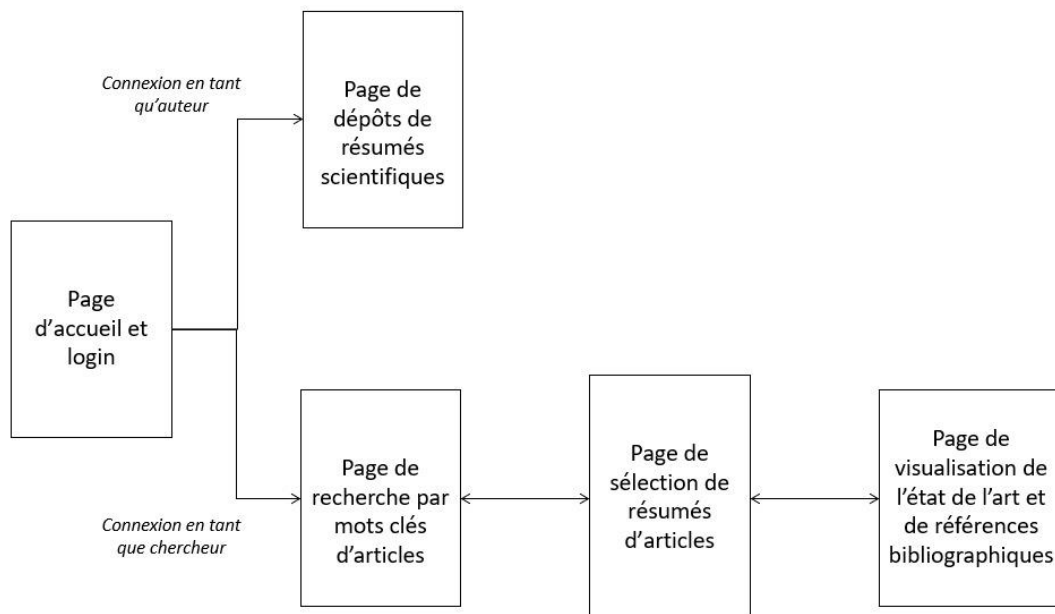
B. Liste des fonctionnalités optionnelles

Les fonctionnalités facultatives sont les suivantes :

- Utiliser l'Intelligence Artificiel afin de détecter le plagiat.
- Générer un résumé automatisé par Machine Learning.
- Héberger notre plateforme sur le Cloud.

III. Organisation de l'application Web

Voici un diagramme présentant l'organisation globale de notre plateforme :



IV. Vue technique du projet

A. Front-end et Back-end

La réalisation du site web se décompose en deux grandes parties avec : une partie « **cliente** » (le visuel ou « front-end ») et une partie « **serveur** » (le « back-end »).

Le front-end se concentra sur les aspects visuels d'un site Web. Il s'agit de la partie que les utilisateurs voient et avec laquelle ils interagissent. Sur notre plateforme, il s'agira de la consultation des articles, des résumés et l'édition de l'état de l'art.

Alors que le développement back-end comprend la structure, le système, les données et la logique d'un site. C'est dans cette partie que seront stockés tous les données de nos utilisateurs, ainsi que tous les articles dont l'utilisateur voudra générer l'état de l'art.

La communication entre les deux parties permettra une utilisation optimale de la plateforme pour l'utilisateur.

B. Outils Utilisés

Pour le **Front-End**, nous utiliserons principalement le framework **Angular** basé sur TypeScript. Son principal intérêt est de rendre l'expérience de l'utilisateur sur les applications web beaucoup plus agréables avec une navigation fluidifiée.

Pour le **Back-End**, nous utiliserons **Spring** (Node.js) qui est un framework permettant une facilité de développement d'applications Web car il fournit tout ce dont un développeur a besoin pour adopter le langage Java. Comme nous aurons besoin d'une base de données, il sera également nécessaire d'utiliser **Spring Data** qui nous permettra d'accéder à des données provenant de diverses sources de données

C. Base de données

Une base de données est nécessaire afin de stocker toutes les données concernant nos utilisateurs ou encore les articles.

Voici un aperçu des tables que nous utiliserons dans la base de données :

1) Table « users »

Lorsqu'un utilisateur s'inscrit sur notre plateforme, ses données sont ajoutées à la base de données et plus spécifiquement, dans la table « users ». C'est cette table qui stocke toutes les données des utilisateurs lors de leur inscription.

mail	first_name	last_name	password	confirm_password	birth_date	Id_orchid	establishment

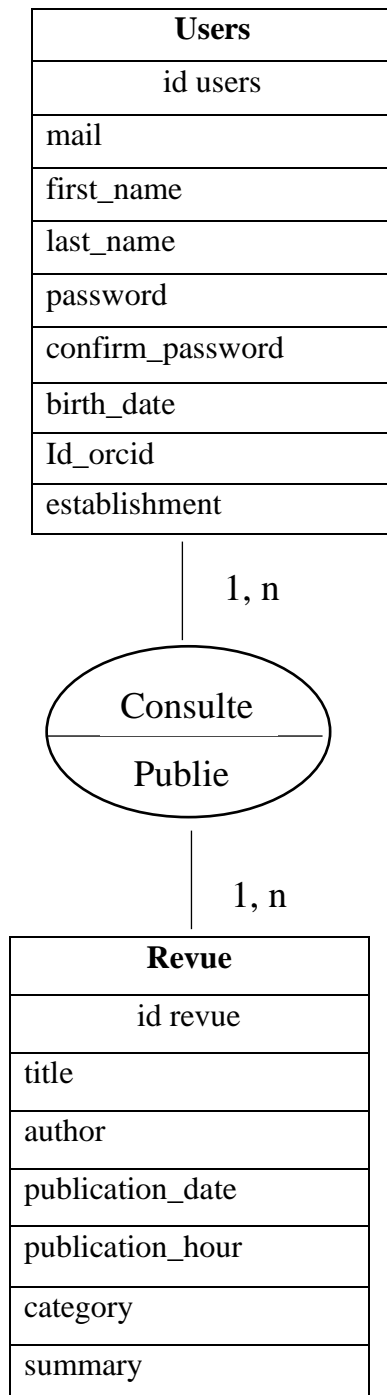
2) Table « revue »

Afin de générer l'état de l'art, il est nécessaire que notre base de données contienne les informations concernant les chercheurs, les revues et les résumés des articles.

title	author	publication_date	publication_hour	category	summary

3) Modèles MCD et MLD

Voici le modèle conceptuel des données ainsi que le modèle relationnel associée :



Users (id_users, mail, first_name, last_name, password, confirm_password, birth_date, id_orcid, establishment)

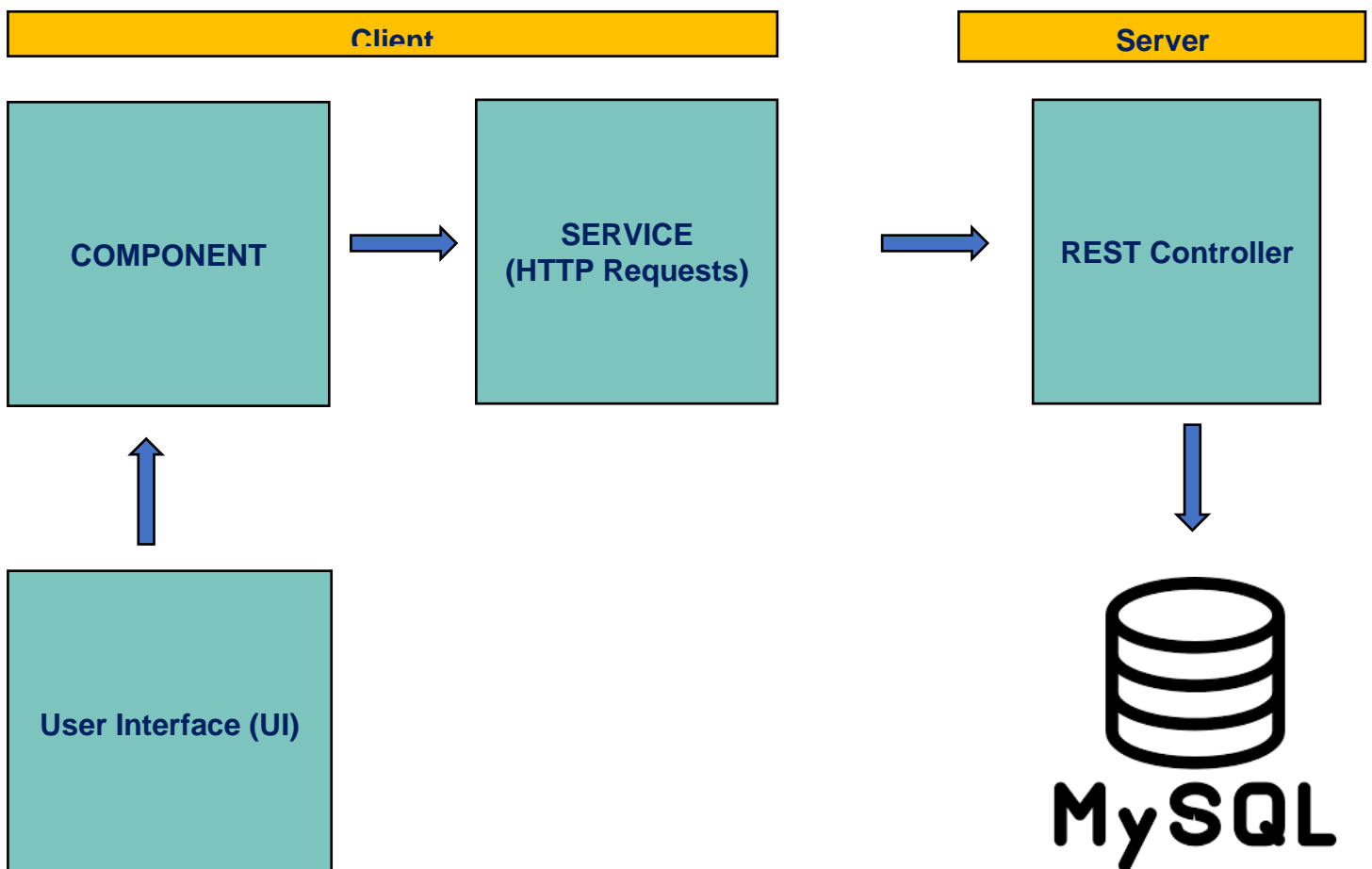
Revue (id revue, title, author, publication_date, publication_hour, category, summary)

D. Design de l'API

Une **API**, pour **A**pplication **p**rogramming **i**nterface, est un programme permettant à deux applications distinctes de communiquer entre elles et d'échanger des données. Cela évite notamment de recréer et redévelopper entièrement une application pour y ajouter ses informations.

Dans notre cas, l'API a donc un rôle essentiel, c'est lui qui permet la communication entre le front end (la partie cliente) et le back end avec la partie serveur.

API Design



E. Angular

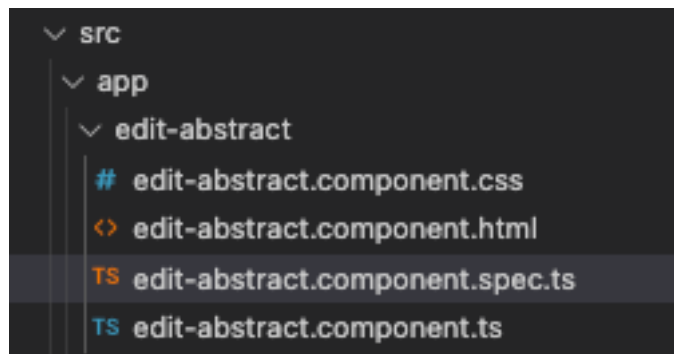
Il faut savoir que Angular permet de créer une multitude de brique logiciel (composant, service, classe...) que l'on va assembler entre eux pour créer notre application web. Chaque section étant indépendante, il est plus simple de faire évoluer notre application web sans que cela ne touche les autres parties du projet.

1) Les composants

Les composants sont le principal élément de base des applications angulaires. Chaque composant est composé de :

- *comp1.component.html* : un modèle HTML qui déclare ce qui s'affiche sur la page.
- *comp1.component.css* : un sélecteur CSS qui est la feuille de style de notre component.
- *comp1.component.spec.ts* : une classe TypeScript qui permet de réaliser des tests.
- *comp1.component.ts* : c'est le contrôleur de notre component. Il va nous permettre d'y ajouter des fonctionnalités entre la vue, les boutons, la base de données, etc.

Voici un exemple de composant pour la page d'édition des abstracts :



Des composants du même type seront alors créer pour les autres fonctionnalités de notre application web :

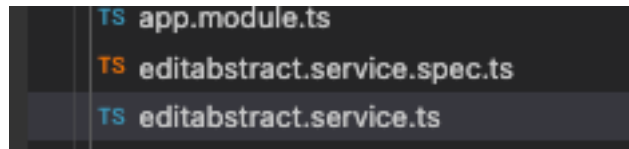
- “**user-login**” pour la connexion des utilisateurs;
- “**user-register**” pour l'inscription des utilisateurs;
- “**abstract-search**” pour la recherche des articles;
- ...

2) Les services

Un autre élément principal du framework Angular est le service. En effet, les services permettent une communication entre les composants.

Dans notre cas, cela permettra la communication entre le front-end et le back-end : lorsque l'utilisateur soumettra un formulaire, les données seront récupérées pour les envoyer dans le back-end. Cette action est donc possible grâce aux services.

Ainsi à chaque composant créé, un service y sera associé.



```
TS app.module.ts
TS editabstract.service.spec.ts
TS editabstract.service.ts
```

On voit par exemple dans ce service que la communication des données se déroule via le localhost:8081 :



```
9  export class EditabstractService {
10
11    baseUrl="http://localhost:8081/summary"
12    constructor(private httpClient:HttpClient) { }
13
14    submitAbstract(abstract:Abstract):Observable<Object>{
15      console.log(abstract);
16      return this.httpClient.post(`${this.baseUrl}`,abstract)
17    }
18  }
```

Des services du même type seront alors créés pour les autres fonctionnalités de notre application web :

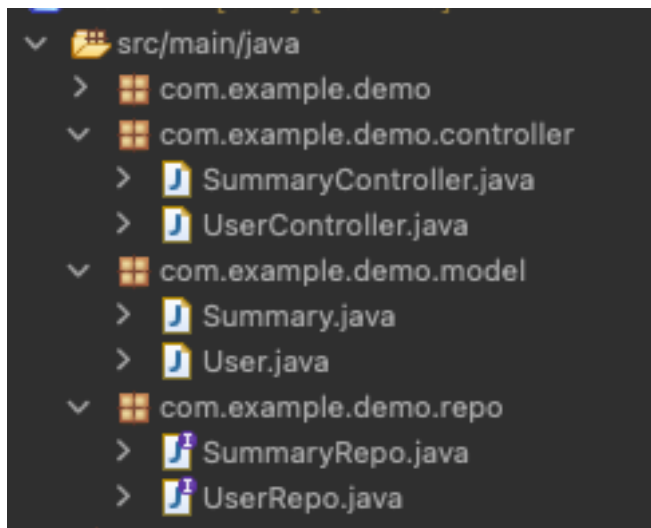
- **“userlogin.service”** pour la connexion des utilisateurs;
- **“userregister.service”** pour l’inscription des utilisateurs;
- **“abstractsearch.service”** pour la recherche des articles;
- ...

F. Spring Boot

Pour notre application web, la nécessité d'interagir avec des données externes est essentiel (par exemple une base de données, un autre programme, ou même le système de fichiers).

De ces différents besoins, une architecture en couches a émergé, avec un rôle pour chaque couche :

- **couche Controller** : gestion des interactions entre les utilisateurs et l'application ;
- **couche Repository** : interaction avec les sources de données externes ;
- **couche Model** : implémentation des objets métiers qui seront manipulés par les autres couches.



Ainsi pour chacune des fonctionnalités, on créera un controller, un fichier repository et un fichier model.

Controller :

- SummaryController.java
- LoginUserController.java
- RegisterUserController.java
- SearchAbstractController.java
- SelectAbstractController.java
- DisplayRevueController.java
-

Model :

- Summary.java
- LoginUser.java
- RegisterUser.java
- SearchAbstract.java
- SelectAbstract.java
- DisplayRevue.java
-

Repository :

- SummaryRepo.java
- LoginUserRepo.java
- RegisterUserRepo.java
- SearchAbstractRepo.java
- SelectAbstractRepo.java
- DisplayRevueRepo.java
-