Présentation finale

Vectorisation de texte

Hanane, Farouq, Walid, Asma, Steve

Université d'Avignon

6 janvier

AVIGNON UNIVERSITÉ

Plan

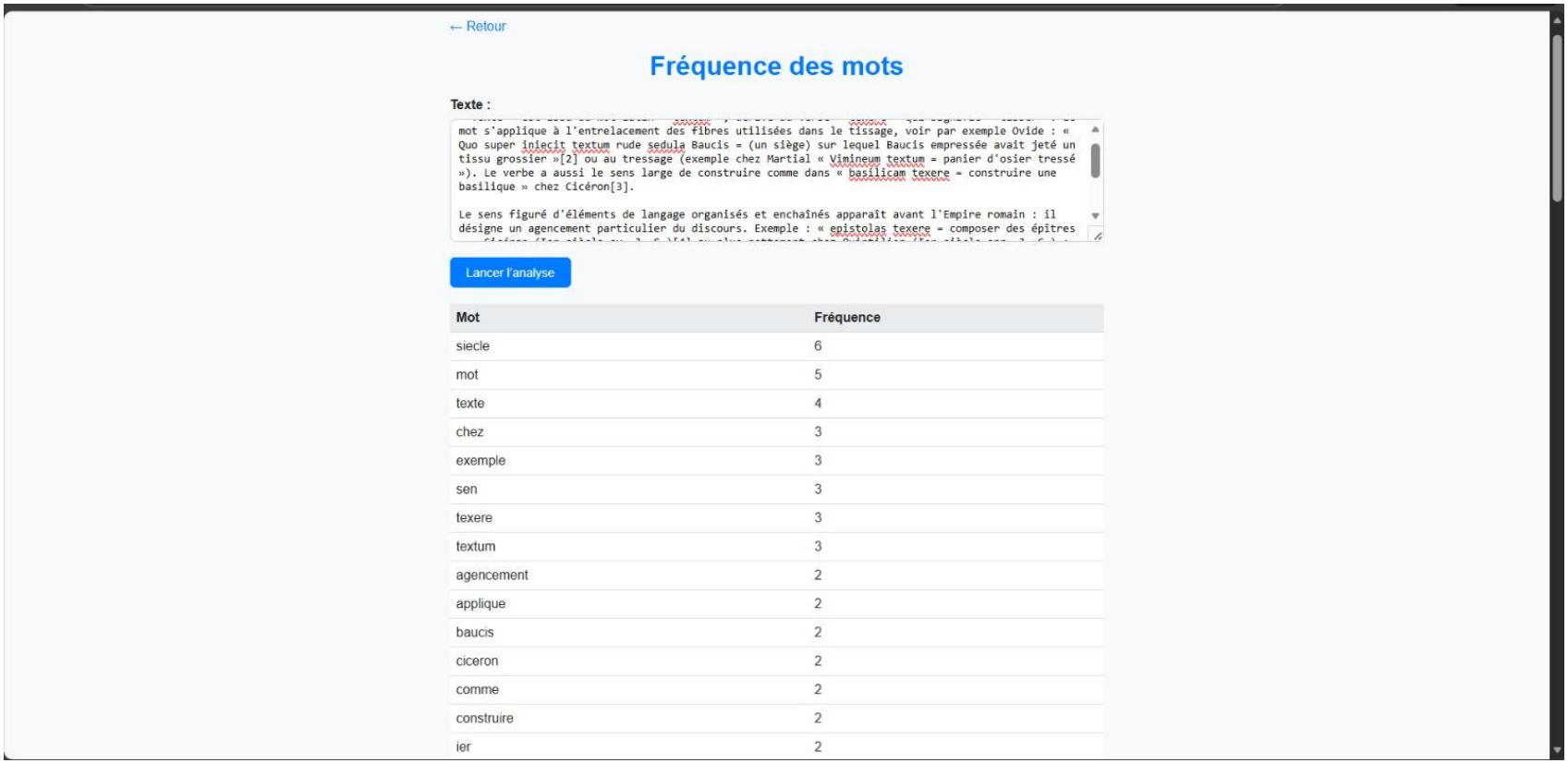
1 Démo finale

2 Projet et méthode Scrum

3 Bilan

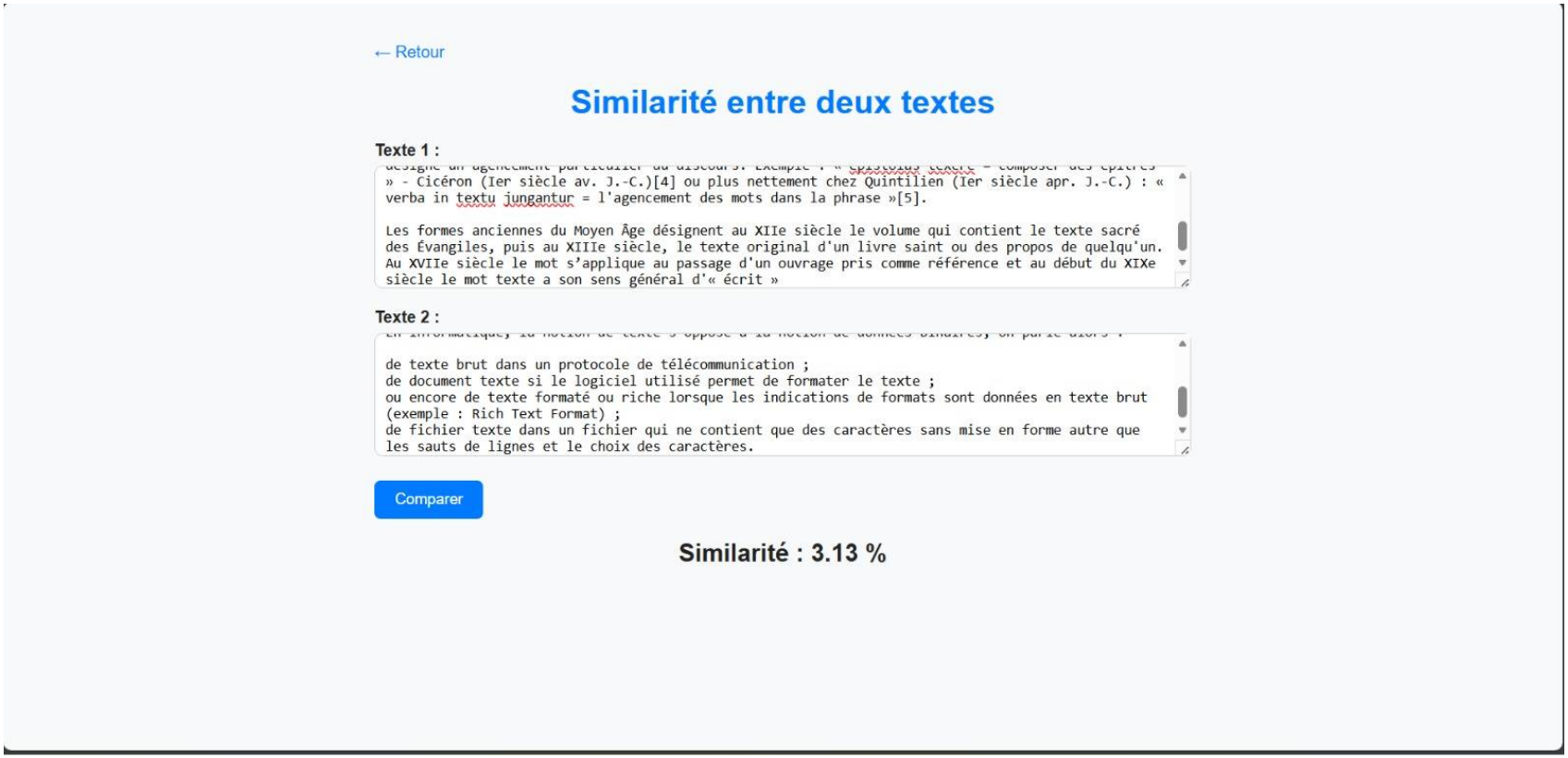
US2 – Fréquence des mots

- On colle un texte.
- On affiche les mots les plus fréquents.



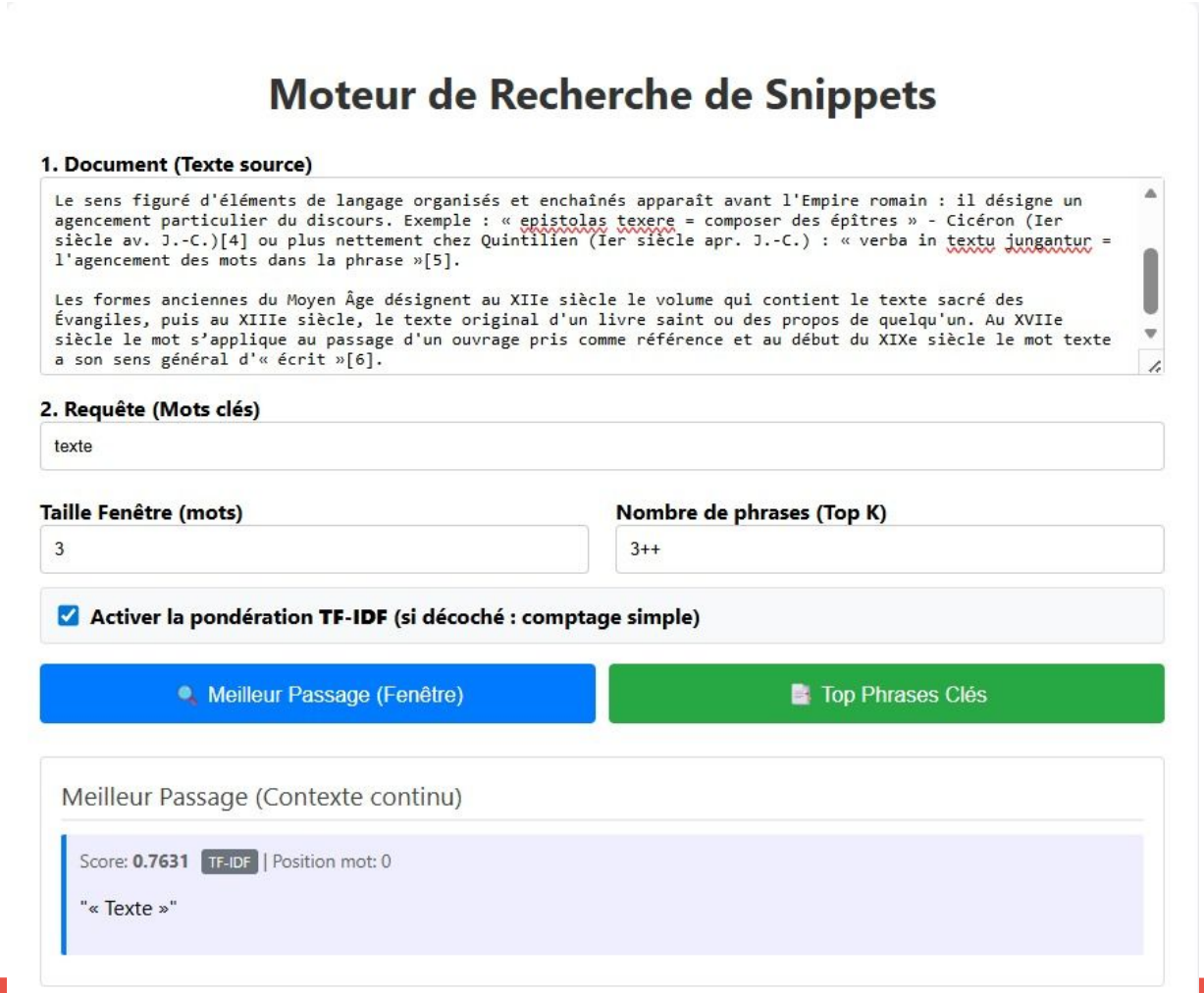
US4 – Similarité entre deux textes

- On colle deux textes.
- On obtient un score de similarité.



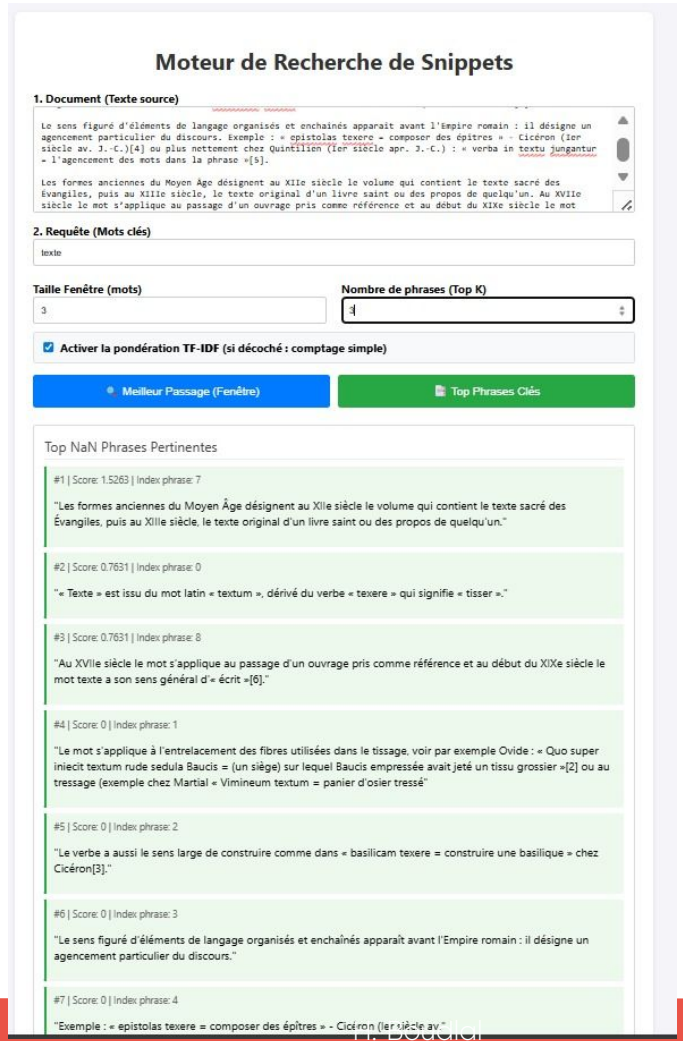
US6 – Snippets : meilleur passage

- On donne un document + une requête.
- On affiche le meilleur passage (fenêtre).



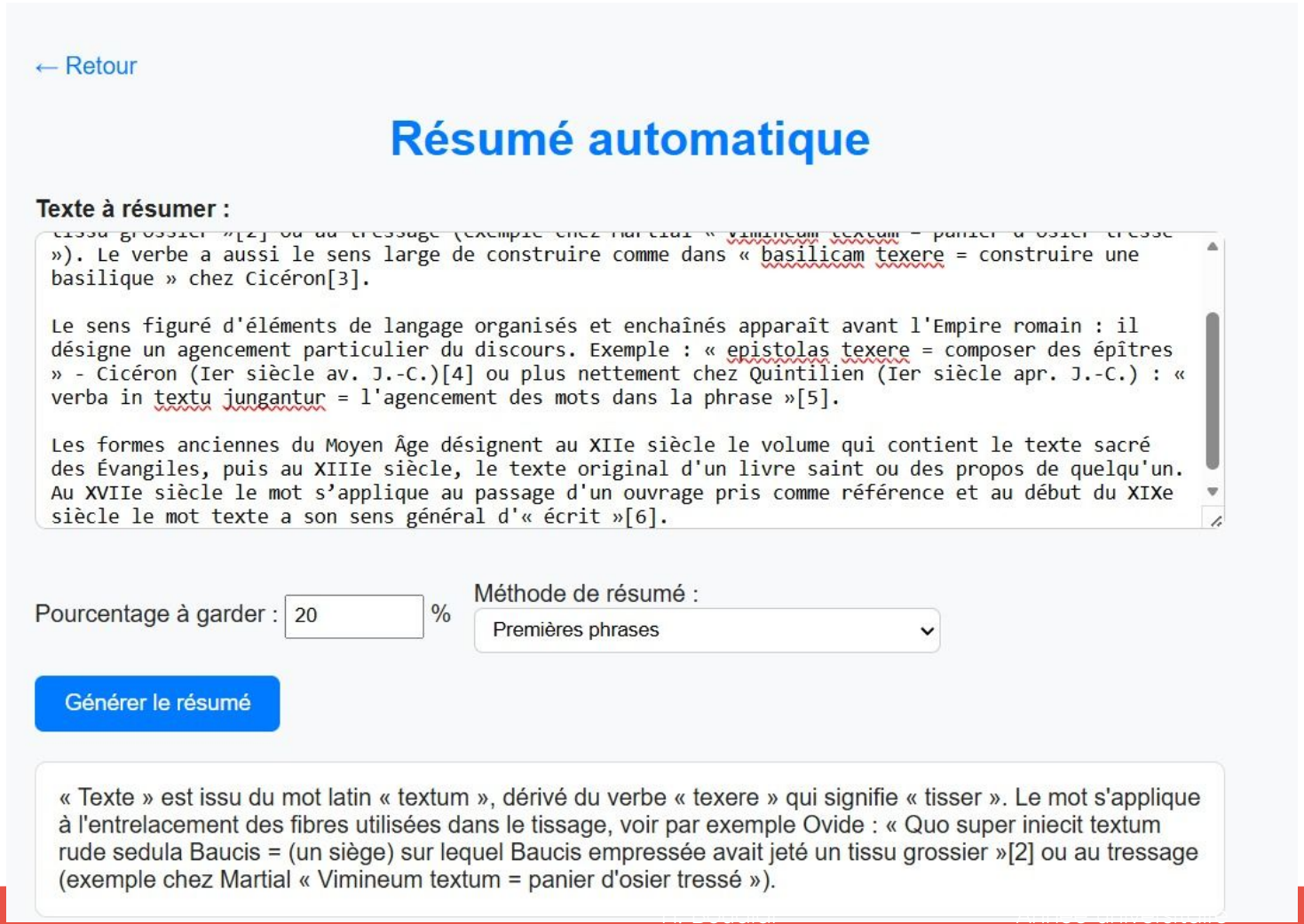
US6 – Snippets : Top phrases (Top K)

- On choisit Top K.
- On affiche les phrases les plus pertinentes.



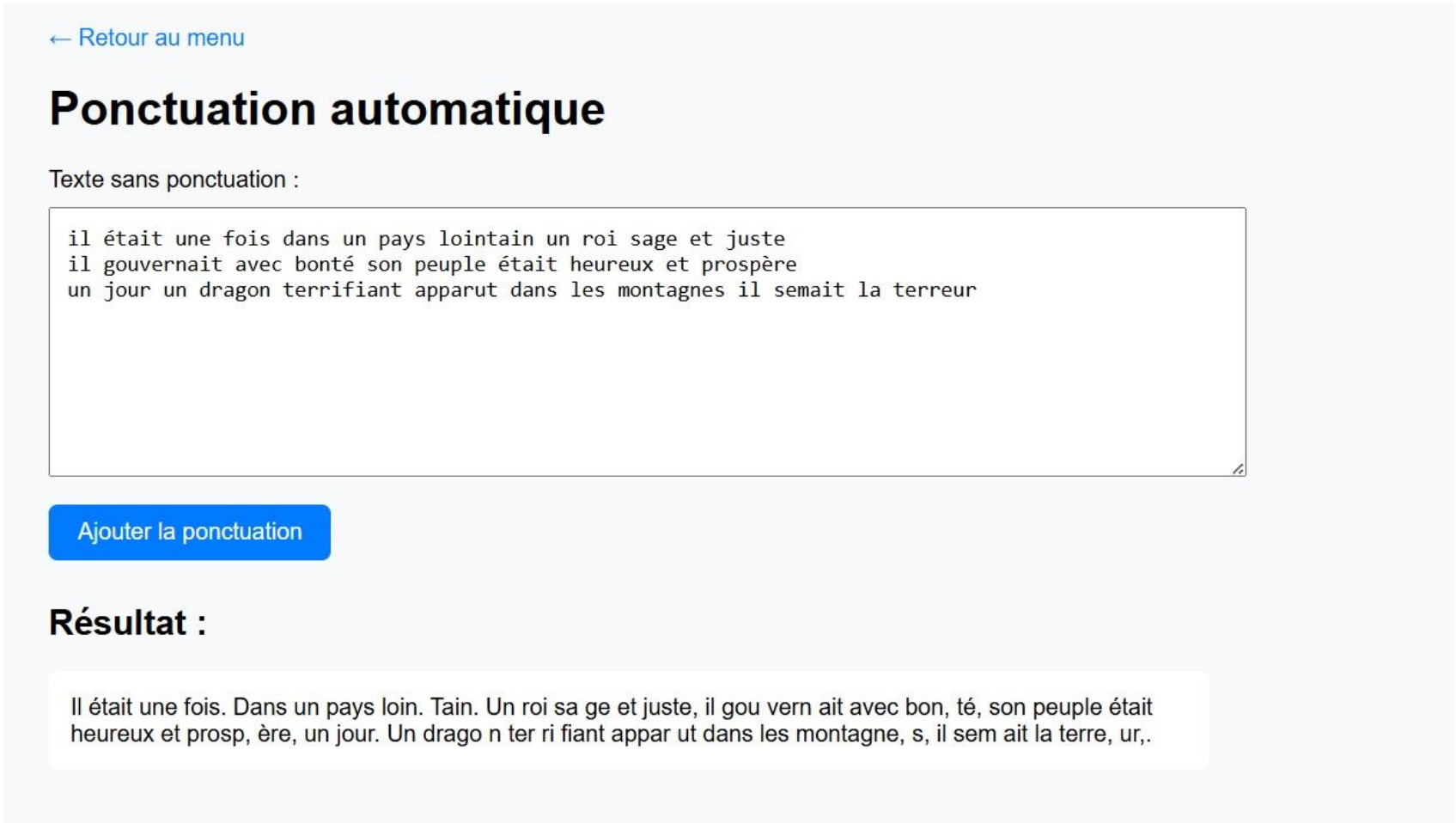
US5 – Résumé automatique

- On choisit un pourcentage à garder.
- On génère le résumé avec une méthode.



US7 – Ponctuation automatique

- On donne un texte sans ponctuation.
- On obtient une version plus lisible.



Sujet du projet

- Thème : application web de traitement de texte.
- On a développé : TF-IDF, similarité, résumé, snippets, ponctuation.
- Objectif : livrer petit à petit en appliquant Scrum.

Product Backlog

- On a écrit nos besoins en **User Stories** : interface, TF-IDF, similarité, résumé, snippets, ponctuation, idée LLM.
- Pourquoi : pour **prioriser** ce qui est important.
- Exemple : l'US **LLM** est restée *à faire* car on a d'abord livré les modules de base.

D	User Story	Definition of Done (DoD	Description / Fonctionnalité	Sprint	Statut
US1	En tant qu'utilisateur, je veux une page d'accueil web claire avec les 4 modules	- Page responsive- 4 boutons fonctionnels- Navigation fluide	Interface de navigation pour rentrer le texte à résumer + la taille , un bouton pour afficher un les mots les plus fréquents en haut dans le texte rentré	1,2,3,4	In progress
US2	En tant qu'utilisateur, je veux afficher les vecteurs de mots (mots les plus fréquents en haut)	- Upload texte- Calcul fréquence- Top 10 mots affichés	Visualisation statistique des mots importants	1	Fait
US3	En tant qu'utilisateur, je veux calculer de TF et IDF	- Saisie texte- Calcul TF/IDF- Affichage tableau	- Saisie texte- Calcul TF/IDF- Affichage tableau		Fait
US4	En tant qu'utilisateur, je veux comparer deux textes pour détecter une similitude / plagiat	- 2 zones texte- Score 0-1- Visualisation	Calcul de similarité entre textes	2,3	Pas commencé
US5	En tant qu'utilisateur, je veux coller un texte et obtenir un résumé	- Choix % réduction- Résumé cohérent- Mots-clés conservés	Module de résumé automatique avec choix de la taille du résumé	2, 3	Pas commencé

Sprints

- On a fait **6 sprints** au total.
- Un sprint = **une séance de TP** (sprint court).
- Pourquoi : en 3h, on vise un **résultat visible** (page, score, module utilisable).

Sprint Backlog + Kanban

- Au début du sprint : on choisit les US du sprint.
- On les découpe en **tâches** et on les met sur un tableau **Kanban**.
- On attribue les tâches selon la difficulté et les capacités de chacun.

timing

Nouveau

Aa taches	Date	hana	État
rectification de TF-IDF et des stopwor	4 décembre 2025 16:00 → 17	H Hanan Boud	Terminé
resumé les 3 methodes	4 décembre 2025 16:00 → 17	A Asma Nihal Boukraa S	Terminé
page du résumé + affichage du résult	4 décembre 2025 16:00 → 17	F FARAH Mohamed Walid	Terminé

Méthodes d'équipe

- **Complexity matching (Sprint 3)** : tâches NLP difficiles aux personnes les plus à l'aise.
- **Pair programming** : utile sur similarité / résumé (Driver–Navigator).
- But : réduire les erreurs et livrer plus vite dans un sprint court.

Rétrospective : déroulement du projet

- Début : on a sous-estimé certaines tâches (ex. TF-IDF Wikipédia).
- Milieu : meilleure organisation (répartition + binômes) ⇒ meilleure stabilité.
- Fin : on a consolidé snippets + ponctuation.

Difficultés rencontrées

- Estimation : TF-IDF Wikipédia plus long que prévu.
- Intégration front/back : calcul OK mais affichage pas prêt (page/menu).
- Tests : peu de tests bout-en-bout (temps de séance).
- Organisation : absences et répartition parfois déséquilibrée.

Leçons apprises

- Garder 10 min fin de sprint : **merge + test complet + mini démo**.
- Une DoD claire aide : *affiché et testable* > *juste calculé*.
- Complexity matching + pair programming : bon choix pour livrer vite.

Perspectives

- Stabiliser l'app : version propre et intégrée (pages + modules).
- Ajouter des tests simples bout-en-bout : texte → résultat affiché.
- Améliorer la ponctuation