



Faculté des Sciences Agadir

Master d'excellence-Analytique des données et intelligence artificielle

Module :

Systèmes Répartis et Distribués

Projet N°7

Moteur de recherche distribué

Réalisé par :

MOUSTIKE Imane

KINAD Kawtar

KHAIR Latifa

KENNOUZ Ayoub

Encadré par :

Pr. IDRAIS Jaafar

Pr. RIDA Zakaria

Année universitaire : 2024-2025



Abstract

À l'ère du Big Data, la gestion efficace de vastes volumes de données hétérogènes constitue un défi majeur. Ce projet s'inscrit dans cette problématique en proposant la conception et l'implémentation d'un moteur de recherche distribué basé sur la stack ELK (Elasticsearch, Logstash, Kibana), complété par une interface web développée avec Flask. L'objectif principal est de fournir un système capable d'ingérer, d'indexer, de rechercher et de visualiser de grandes quantités de données textuelles issues de formats variés (CSV, JSON, API REST). Grâce à une architecture distribuée, ce moteur offre des performances accrues, une scalabilité horizontale et une haute disponibilité. Le projet inclut également la création de dashboards interactifs via Kibana pour explorer les tendances en intelligence artificielle. Les résultats obtenus démontrent l'efficacité de cette approche dans le traitement de données massives tout en assurant une expérience utilisateur fluide et intuitive.

Table des matière

Introduction générale.....	1
Architecture Technique	3
1 Description générale de l'architecture.....	3
1.1 Vue d'ensemble	3
1.2 Type de communication	4
1.3 Enchaînement des traitements – Cycle complet de recherche	4
1.3.1 Phase d'indexation (préparation des données)	5
1.3.2 Phase de recherche	6
Diagrammes UML.....	7
2 Diagramme de Cas d'Utilisation.....	7
3 Diagramme de Classes	8
4 Diagramme de Séquence	10
5 Diagramme de Déploiement.....	11
6 Diagramme d'Activité	12
Implémentation et Réalisation.....	13
7 Mise en œuvre du système distribué	13
7.1 Ingestion et Indexation des Données.....	13
7.2 Interface de Recherche (Flask).....	13
7.3 Dashboards et Visualisation (Kibana)	16
Conclusion générale	22
Références	23

Liste des Figures

Figure 1:l'architecture globale du moteur de recherche distribué avec ELK	4
Figure 2: Enchaînement des traitements – Cycle complet de recherche.....	5
Figure 3:Diagramme de Cas d'Utilisation	7
Figure 4:Diagramme de Classes.....	9
Figure 5:Diagramme de Séquence	10
Figure 6:Diagramme de Déploiement	11
Figure 7:Diagramme d'Activité.....	12
Figure 8: La liste des index dans Kibana	13
Figure 9:Interface de recherche.....	14
Figure 10: Interface "À propos" du moteur de recherche distribué	14
Figure 11 :Recherche par mot-clé : “Apach”	15
Figure 12:Recherche par mot-clé : “Hacker news ”	15
Figure 13:Recherche par mot-clé : “pyhton ”	16
Figure 14:Recherche par tags.....	16
Figure 15:Exploration visuelle des actualités et tendances en intelligence artificielle	17
Figure 16: Tableau de bord Kibana : Analyse globale des projets en intelligence artificielle .	18
Figure 17:Tableau de bord Kibana : Exploration des articles Hacker News sur l’intelligence artificielle	19
Figure 18:Tableau de bord Kibana : Analyse interactive des projets en intelligence artificielle	19
Figure 19:Tableau de bord Kibana : Exploration interactive des chatbots de la Chatterbot Collection	20
Figure 20:Tableau de bord Kibana : Analyse interactive des incidents de violation de données	21

Liste des tableaux

Tableau 1: les modules utilisés.....	3
--------------------------------------	---

Introduction générale

À l'ère du Big Data, où plus de 328,77 millions de téraoctets de données sont générés chaque jour (Statista, 2024), les solutions de recherche centralisées révèlent des limites significatives en termes de performance et de scalabilité. Les architectures traditionnelles rencontrent des difficultés pour traiter des requêtes complexes sur des volumes massifs de données, entraînant des temps de réponse excessifs (dépassant souvent 1 seconde pour des corpus de plus de 10 millions de documents) et des risques accrus de défaillance.

Face à ces contraintes, **les systèmes distribués** s'imposent comme une solution optimale. En répartissant le traitement des données sur plusieurs nœuds, ces architectures offrent des avantages majeurs : une latence réduite (moins de 200 ms même sous forte charge), une scalabilité horizontale (permettant l'ajout transparent de nouvelles ressources) et une haute disponibilité grâce à la réplication automatique des données.

Ce projet vise à concevoir et implémenter **un moteur de recherche distribué** capable de traiter efficacement de vastes volumes de données textuelles provenant de sources variées (fichiers CSV, JSON, APIs REST). Basé sur la suite **ELK** (Elasticsearch, Logstash, Kibana), ce moteur intègre une interface utilisateur simplifiée permettant d'effectuer des recherches par mots-clés, filtres ou tags. L'objectif est d'offrir une solution performante et évolutive pour répondre aux besoins croissants en matière d'analyse et d'exploitation de données.

Le projet s'appuie sur plusieurs technologies clés :

- **Elasticsearch 9.0.0** pour l'indexation et la recherche distribuée.
- **Logstash 9.0.0** pour l'ingestion et la transformation des données.
- **Kibana 9.0.0** pour la visualisation des données et le suivi des performances.
- **Flask** (Python) pour développer une interface web permettant aux utilisateurs d'interroger le système.
- **JSON, CSV et API REST** comme formats principaux pour l'échange et le traitement des données.

Contrairement à d'autres approches reposant sur des protocoles comme RPC, RMI ou CORBA, cette solution privilégie une communication distribuée basée sur des appels REST, caractéristique des architectures modernes orientées données. Ce choix offre plusieurs avantages : simplicité de mise en œuvre, compatibilité avec différents langages de programmation et facilité d'extension lors de l'ajout de nouvelles machines.

Architecture Technique

L'architecture proposée repose sur un ensemble de composants distribués, interconnectés, qui assurent une ingestion efficace des données, une indexation scalable, une recherche rapide et une visualisation intuitive. L'ensemble du système est pensé pour supporter de grandes volumétries tout en restant résilient.

1 Description générale de l'architecture

1.1 Vue d'ensemble

Notre moteur de recherche adopte une **architecture modulaire distribuée** articulée autour de la suite **ELK**. Chaque composant est isolé, interagit par API REST et peut évoluer indépendamment :

Tableau 1: les modules utilisés

Module	Rôle principal	Flux entrant	Flux sortant
Logstash	Ingestion & transformation des données (CSV, JSON/API).	Sources de données hétérogènes.	Documents JSON normalisés vers Elasticsearch.
Cluster Elasticsearch	Stockage distribué, indexation, recherche full-text.	Docs JSON de Logstash, requêtes REST de l'API.	Résultats de recherche, agrégations analytiques.
Kibana	Visualisation & monitoring.	Données indexées (via requêtes Kibana).	Dashboards, graphes..
API Flask	Couche métier « search ». Expose endpoints /search,	Requêtes HTTP du Frontend.	JSON paginé (hits, scores, facettes).
Frontend Web	Interface utilisateur responsive.	Appels AJAX vers l'API.	Affichage des résultats, filtres dynamiques.

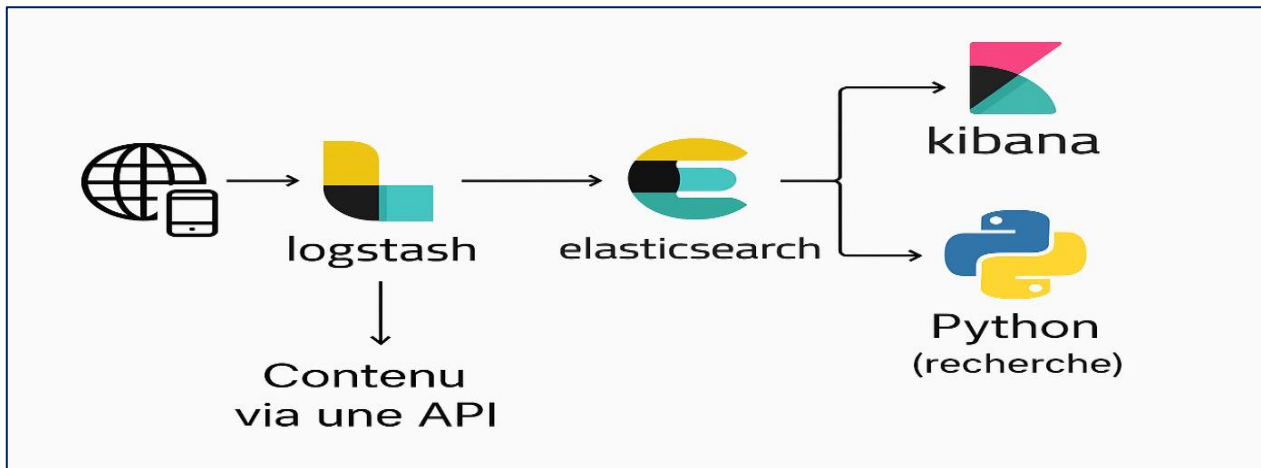


Figure 1: l'architecture globale du moteur de recherche distribué avec ELK

1.2 Type de communication

- **Client ⇄ API Flask:**

Le navigateur envoie des requêtes HTTP (au format JSON) à l'API, exactement comme quand on visite un site web classique.

- **API Flask ⇄ Elasticsearch:**

L'API parle à Elasticsearch par des requêtes HTTP sur le port 9200, donc toujours du simple REST.

- **Logstash ⇄ Elasticsearch :**

Logstash envoie les données préparées à Elasticsearch par un flux HTTP « bulk ».

1.3 Enchaînement des traitements – Cycle complet de recherche

Le moteur fonctionne en deux temps : **indexation** (préparer les données) puis **recherche** (interroger les données).

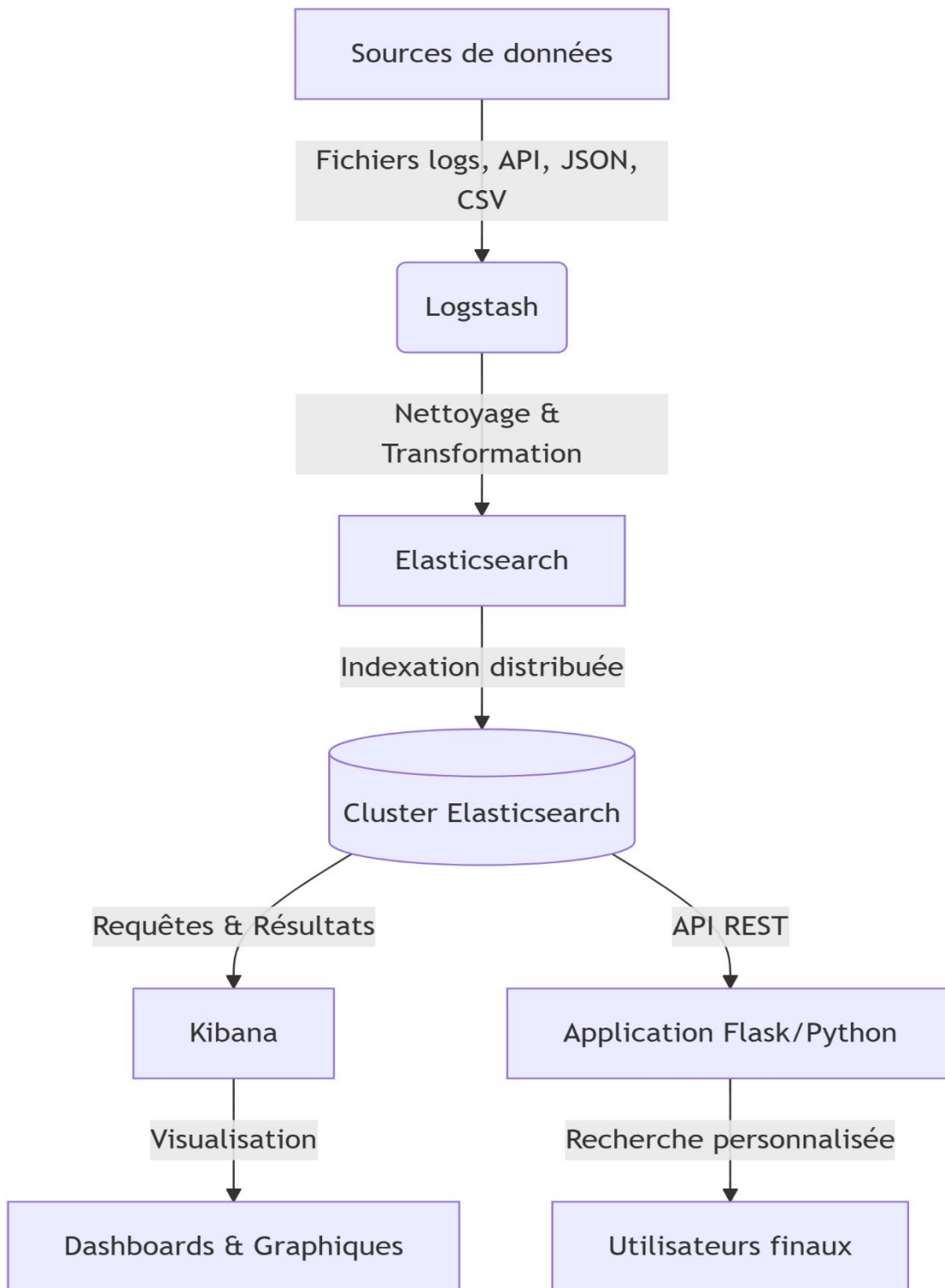


Figure 2: Enchaînement des traitements – Cycle complet de recherche

1.3.1 Phase d'indexation (préparation des données)

Ce processus est lancé automatiquement ou périodiquement :

- **Collecte des données avec Logstash (Serveur d'ingestion) :** Logstash aspire les données :

Fichiers *CSV/JSON* stockés dans un dossier partagé.

Réponses *d'APIs REST* (scraping programmé).

- **Nettoyage & transformation des données au format JSON avec Logstash:** Filtres Logstash : champ “title”, “tags”, “published_at” convertis en JSON propre.
- **Envoi des données à Elasticsearch pour indexation avec des pipelines “bulk” :** Logstash poste des lots de documents sur le port 9200 (vers Elasticsearch).
- **Indexation distribuée avec Elasticsearch (Base de données) :** Dans Elasticsearch répartit chaque document dans des shards. Chaque shard possède une réplique pour éviter de perte si un nœud tombe.
- **Données prêtes – Elasticsearch :** Après ~1 seconde de rafraîchissement, les nouveaux documents deviennent interrogeables.

1.3.2 Phase de recherche

Ce processus est déclenché à chaque fois qu'un utilisateur effectue une recherche :

- **Saisie de la requête – Interface Web (Client)**
 - ✓ Soit dans notre **interface Web**.
 - ✓ Soit directement dans **Kibana Discover**.
- **Transmission -- Client → API Flask (Serveur applicatif) :** La requête est transmise via L'API Flask (Exemple : /search?q=machine+learning&tags=nlp (appel HTTPS))
 - **Analyse de la requête – API Flask :** Traduit les paramètres en Query DSL Elasticsearch
 - **Retour des résultats - API Flask → Client :** L'API renvoie un JSON contenant : titre, extrait surligné, date, score.
 - **Mise à jour des statistiques – API Flask + Kibana :** Les requêtes sont journalisées ; Kibana affiche en temps réel les tops requêtes et la latence moyenne.

Diagrammes UML

2 Diagramme de Cas d'Utilisation

Montre les interactions entre utilisateur, analyste, administrateur et les composants (Kibana, API, Elasticsearch).

- **L'utilisateur** interagit principalement avec l'**API** ou **Kibana** pour soumettre des requêtes et consulter les résultats.
- **L'analyste** consulte les statistiques via **Kibana**.
- **L'administrateur** supervise l'indexation et configure l'ingestion de données.

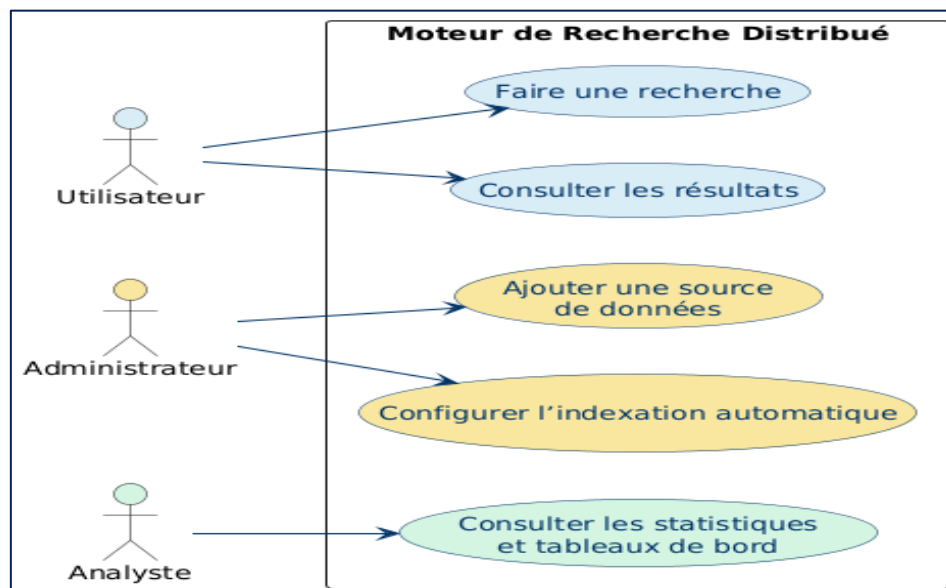


Figure 3: Diagramme de Cas d'Utilisation

3 Diagramme de Classes

Le diagramme de classes décrit la **structure interne du système** : les classes principales, leurs attributs, méthodes et les relations qu'elles entretiennent (association, dépendance...). Il offre une **vue statique** du système.

- **Utilisateur** : interagit avec le système et peut soumettre plusieurs requêtes.
- **RequeteRecherche** : contient le mot-clé saisi et la date ; transmise à l'API pour traitement.
- **API** : fait le lien entre l'utilisateur et Elasticsearch ; envoie les requêtes et génère les statistiques.
- **ElasticsearchCluster** : cœur du moteur, regroupe plusieurs nœuds (NoeudElasticsearch) qui contiennent des Index.
- **Document** : unité de contenu (article, message) stockée dans un index.
- **Logstash** : ingère, nettoie et formate les données avant indexation.
- **Kibana** : visualise les statistiques (fréquence, popularité, performance).

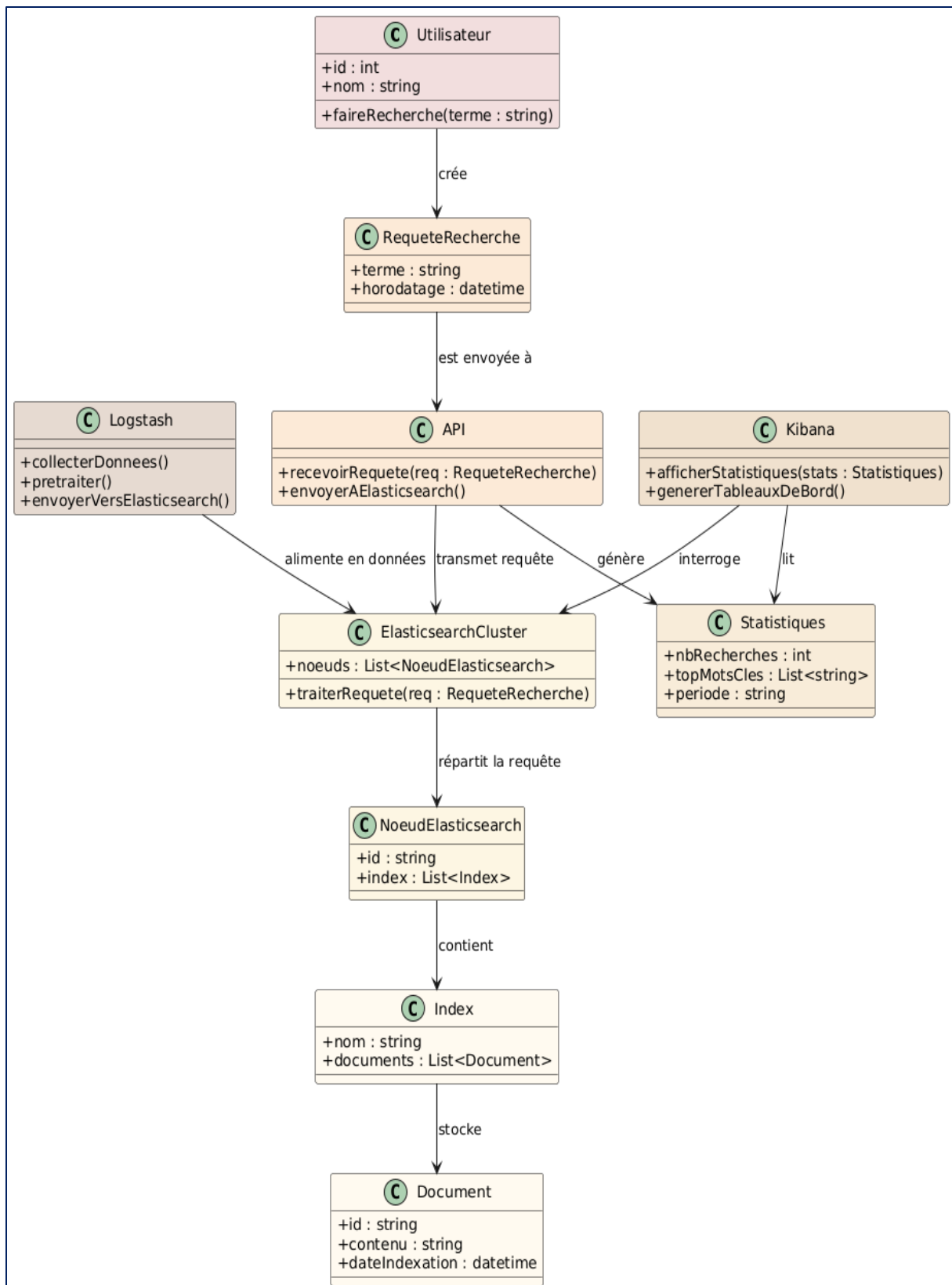


Figure 4: Diagramme de Classes

4 Diagramme de Séquence

Ce diagramme illustre le déroulement d'une recherche utilisateur :

- L'utilisateur saisit une requête via l'interface.
- La requête est transmise à l'API.
- L'API interroge Elasticsearch.
- Elasticsearch renvoie les résultats.
- L'API transmet les résultats à l'interface utilisateur.

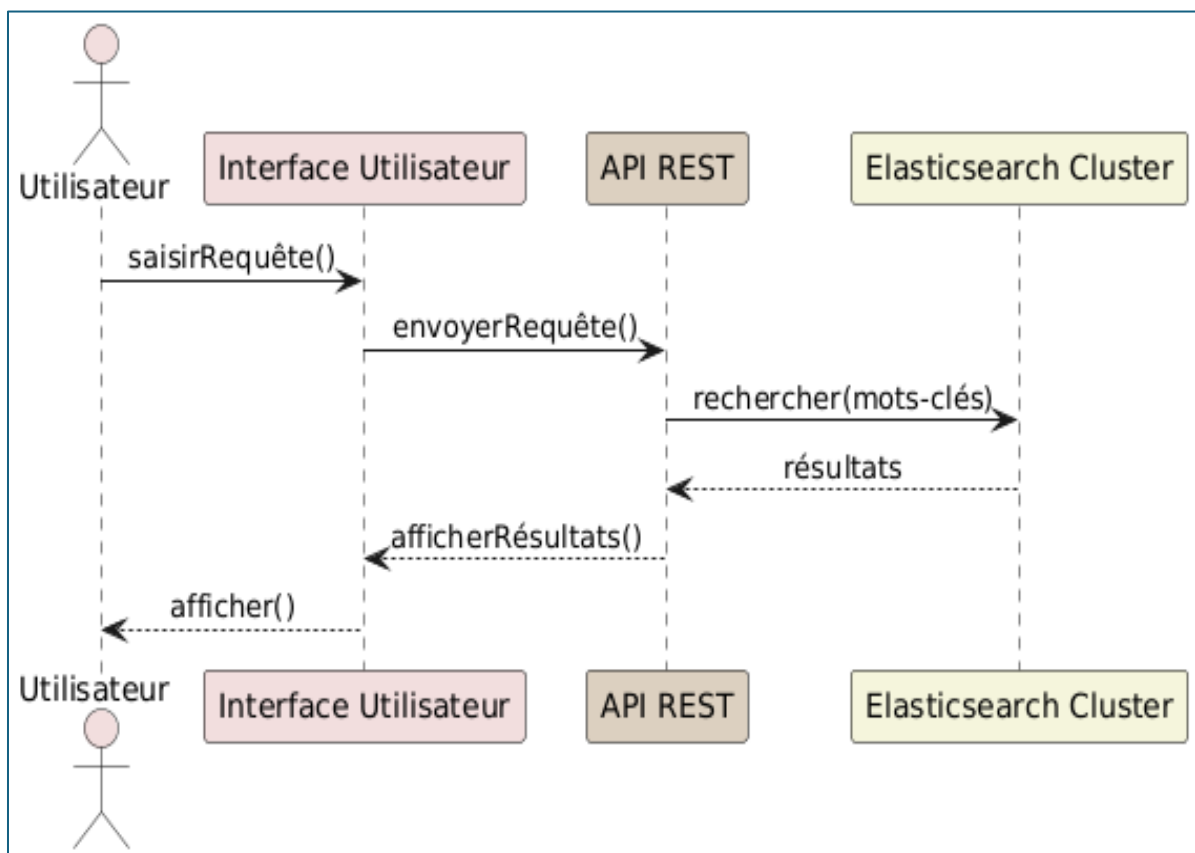


Figure 5:Diagramme de Séquence

5 Diagramme de Déploiement

Ce diagramme montre comment les composants sont distribués dans l'infrastructure :

- **Serveur Logstash** : collecte les données.
- **Serveur API** : gère les requêtes et communique avec Elasticsearch.
- **Serveur Kibana** : interface de visualisation pour les analystes.
- **Cluster Elasticsearch** : composé de plusieurs nœuds, supporte la recherche distribuée.

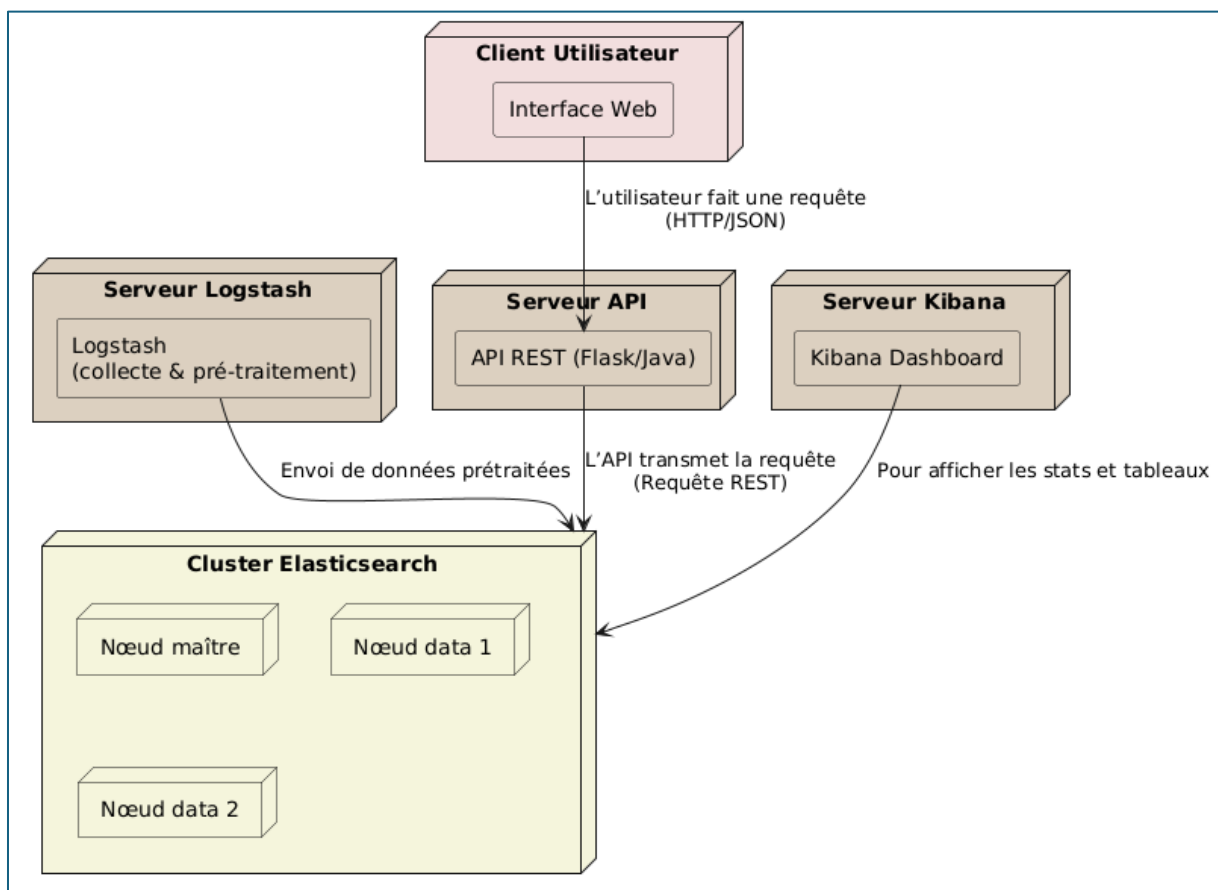


Figure 6:Diagramme de Déploiement

6 Diagramme d'Activité

Ce diagramme détaille les étapes logiques lors d'une recherche :

1. L'utilisateur saisit une requête.
2. L'interface envoie la requête à l'API.
3. L'API transmet à Elasticsearch.
4. Elasticsearch retourne les résultats.
5. L'interface affiche les résultats.

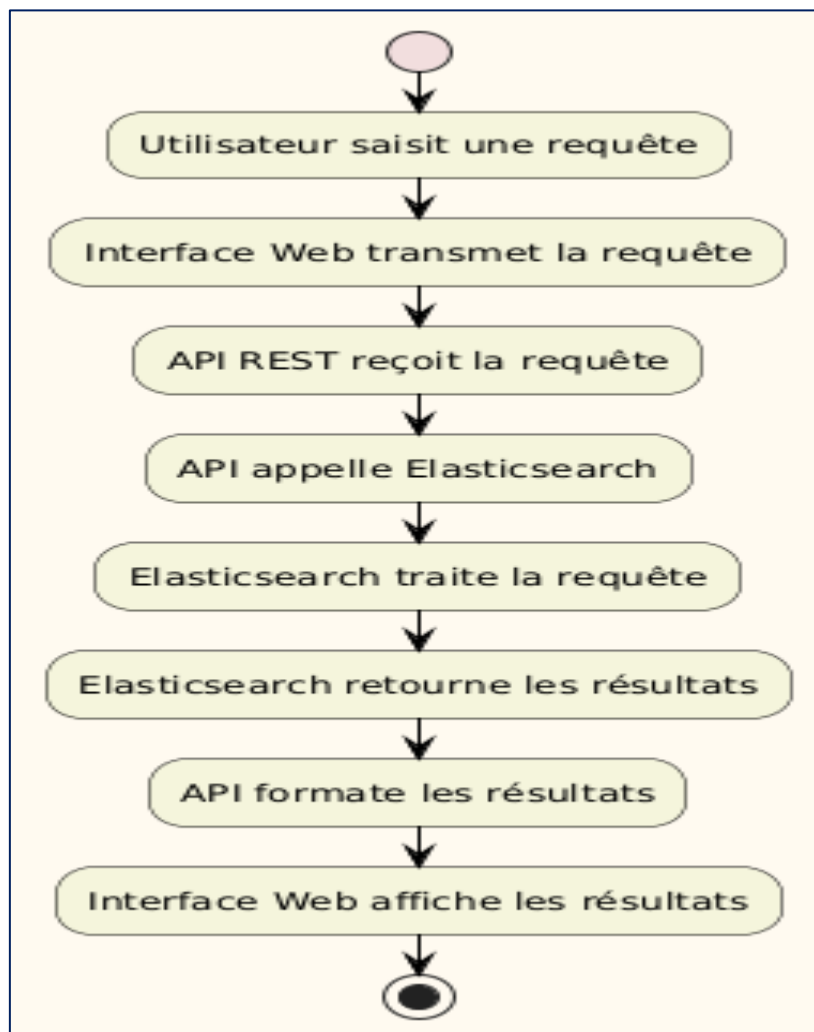


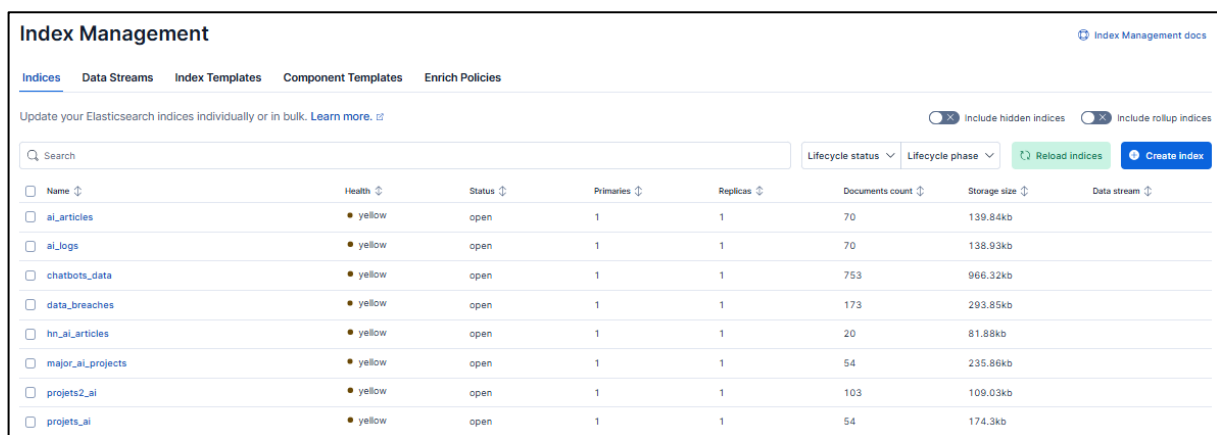
Figure 7:Diagramme d'Activité

Implémentation et Réalisation

7 Mise en œuvre du système distribué

7.1 Ingestion et Indexation des Données

- **Données sources** : fichiers CSV (articles AI/IT), fichiers JSON (issues d'API comme NewsAPI), et fichiers logs.
- Logstash utilise plusieurs pipelines configurés dans des fichiers .conf pour parser, filtrer et envoyer les données à Elasticsearch.
- Plusieurs **index créés** dans Elasticsearch :



The screenshot shows the 'Index Management' page in Kibana. It features a table listing various Elasticsearch indices. The table columns include Name, Health, Status, Primaries, Replicas, Documents count, Storage size, and Data stream. The indices listed are ai_articles, ai_logs, chatbots_data, data_breaches, hn_ai_articles, major_ai_projects, projets2_ai, and projets_ai. All indices have a 'yellow' health status and are 'open'.

Name	Health	Status	Primaries	Replicas	Documents count	Storage size	Data stream
ai_articles	yellow	open	1	1	70	139.84kb	
ai_logs	yellow	open	1	1	70	138.93kb	
chatbots_data	yellow	open	1	1	753	966.32kb	
data_breaches	yellow	open	1	1	173	293.85kb	
hn_ai_articles	yellow	open	1	1	20	81.88kb	
major_ai_projects	yellow	open	1	1	54	235.86kb	
projets2_ai	yellow	open	1	1	103	109.03kb	
projets_ai	yellow	open	1	1	54	174.3kb	

Figure 8: La liste des index dans Kibana

7.2 Interface de Recherche (Flask)

Une API web légère en **Flask** a été développée pour permettre à l'utilisateur de saisir des requêtes de recherche et de recevoir des résultats formatés.

Fonctionnalités offertes :

- Recherche par mots-clés.
- Recherche sur plusieurs champs (titre, description, tags).

- Affichage du titre, description, url et date.

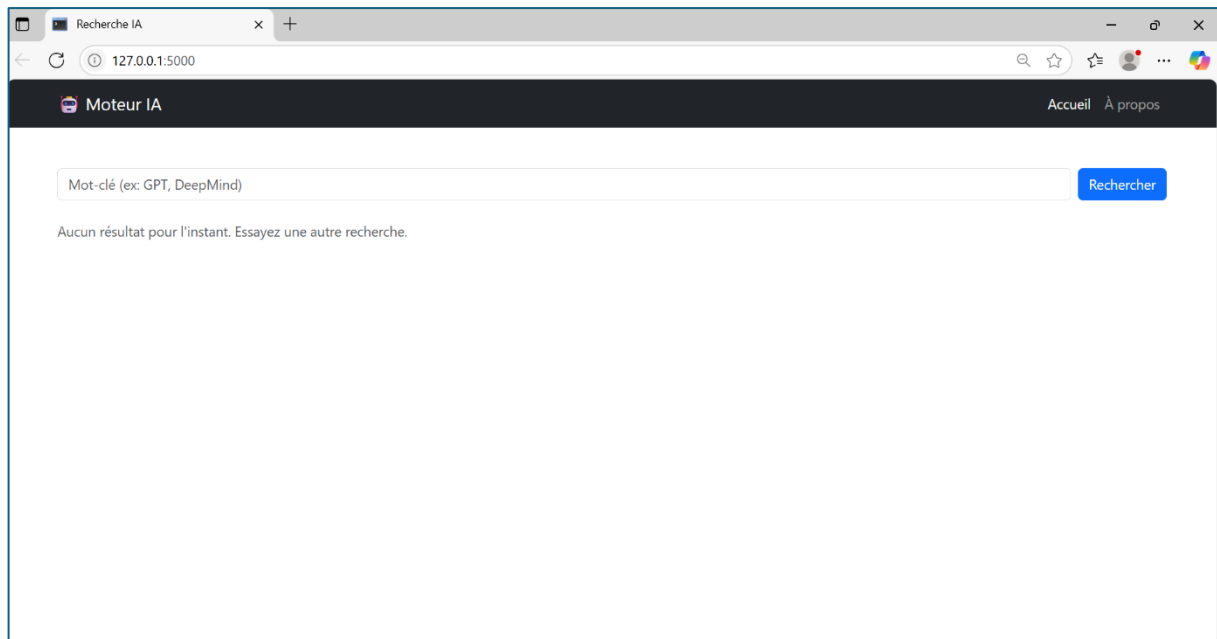


Figure 9: Interface de recherche

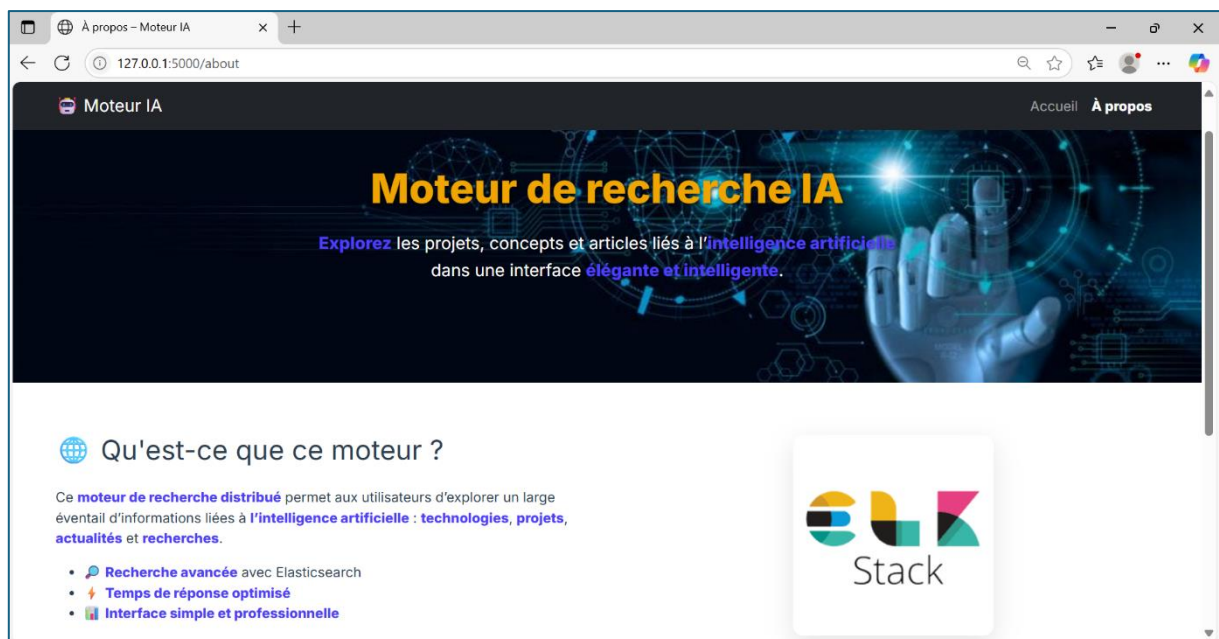


Figure 10: Interface "À propos" du moteur de recherche distribué

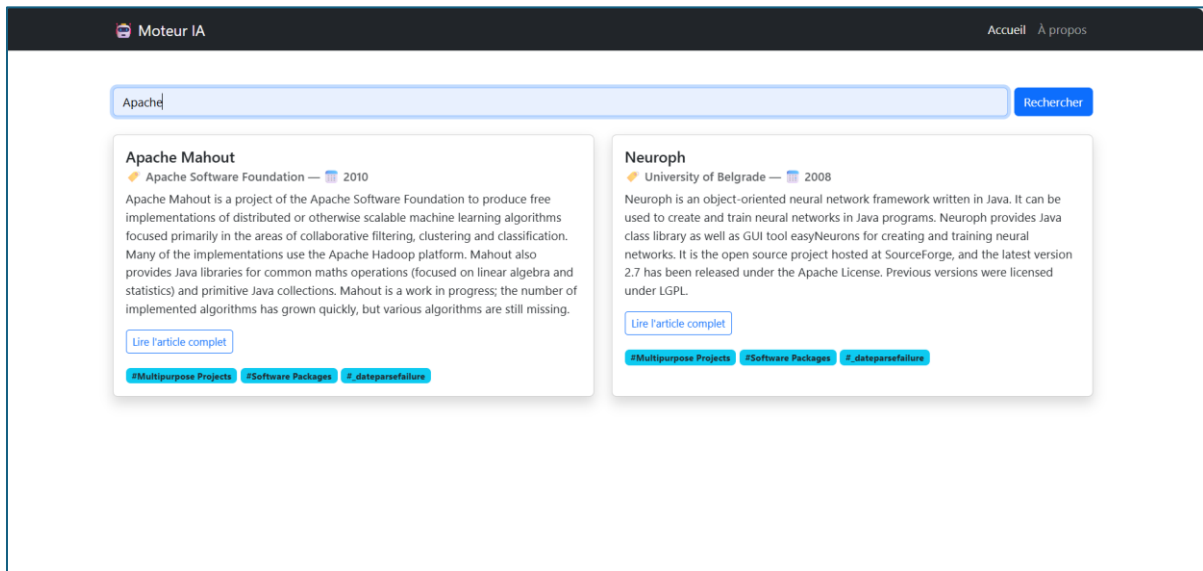


Figure 11 :Recherche par mot-clé : “Apach

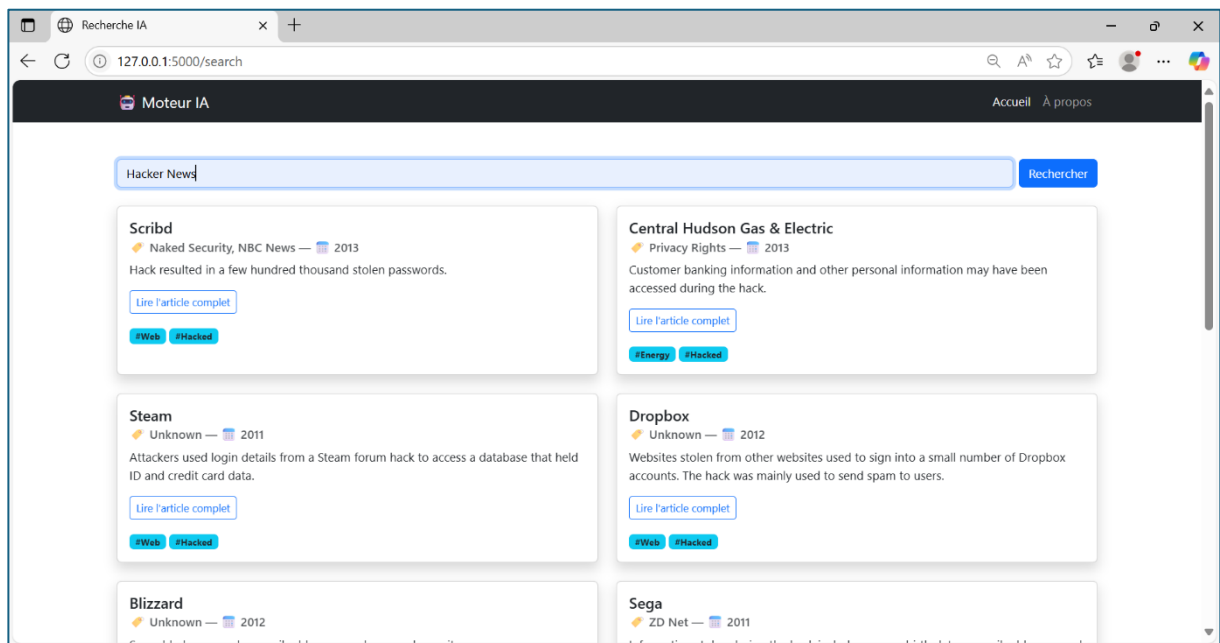


Figure 12:Recherche par mot-clé : “Hacker news ”

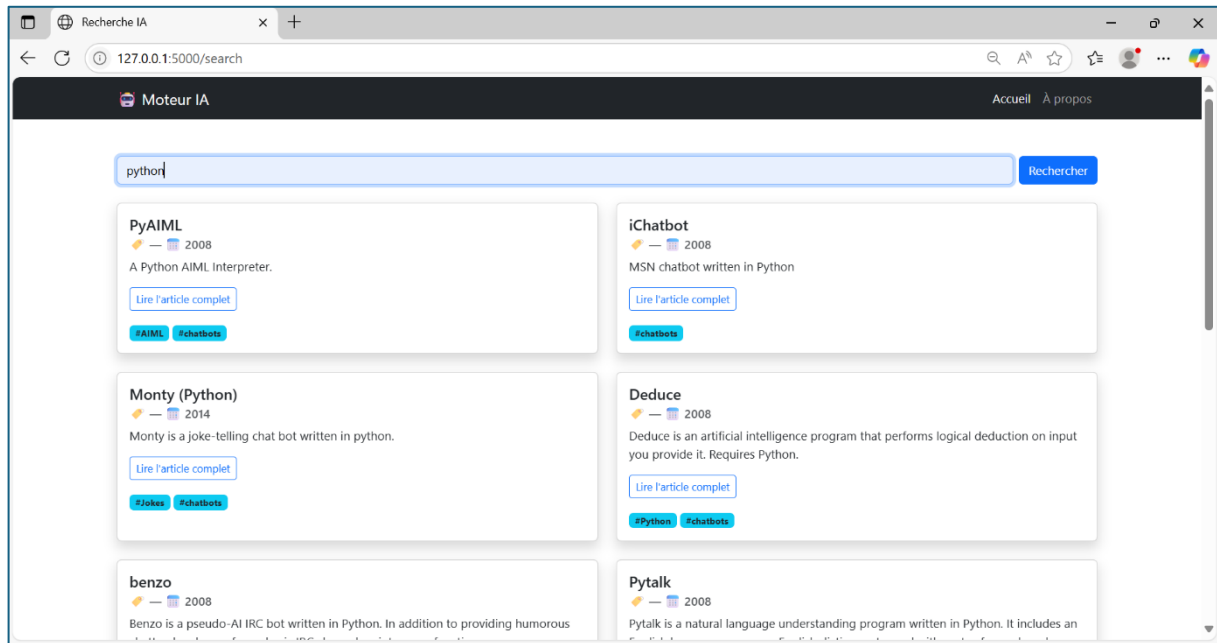


Figure 13: Recherche par mot-clé : “pyhton”

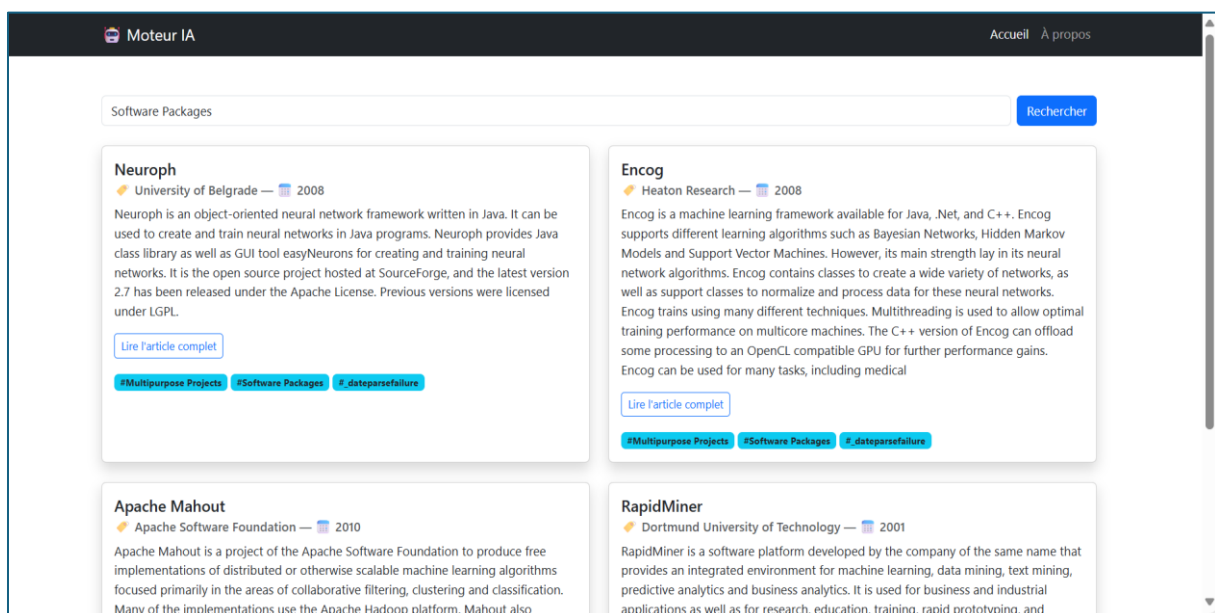


Figure 14: Recherche par tags

7.3 Dashboards et Visualisation (Kibana)

Dans le cadre de cette analyse, nous avons utilisé Kibana, un outil puissant de visualisation de données intégré à la suite ELK (Elasticsearch, Logstash, Kibana). Kibana permet d'explorer, de filtrer et de représenter visuellement des données en temps réel, à travers des tableaux de bord interactifs. Grâce à sa compatibilité avec les formats structurés comme JSON ou CSV, nous avons pu facilement importer notre jeu de données lié aux violations de données et construire des visualisations claires et dynamiques.

Voici les différents dashboards réalisés, chacun mettant en lumière une facette spécifique des incidents répertoriés : leur nature, leur répartition temporelle, les sources, les tags associés, les mots clés des messages, etc. Chacun est accompagné d’une description détaillée.

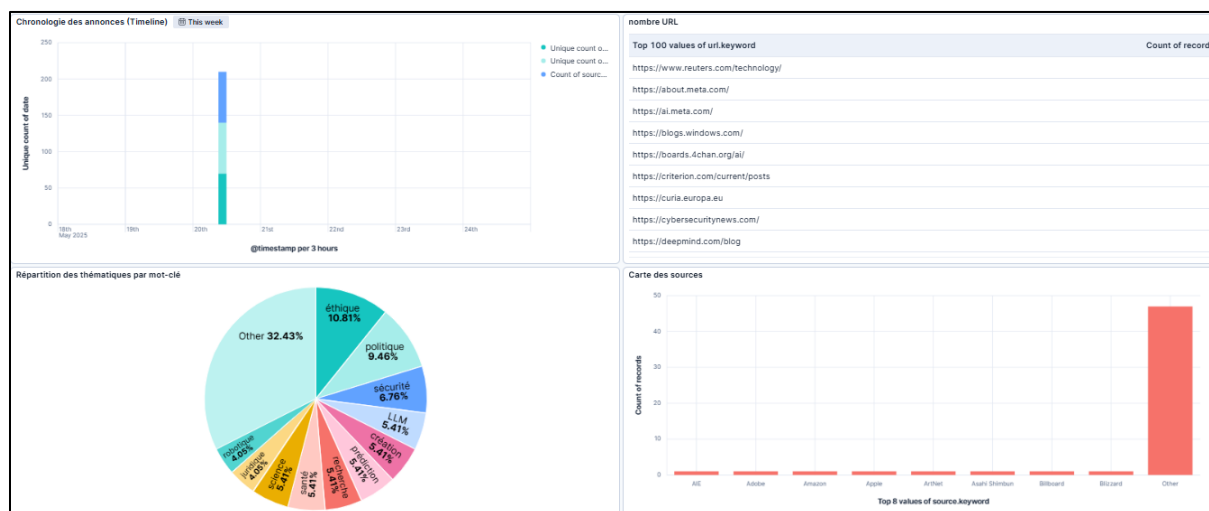


Figure 15: Exploration visuelle des actualités et tendances en intelligence artificielle

Le tableau de bord offre une vue d’ensemble dynamique et interactive de l’actualité de l’intelligence artificielle à travers quatre visualisations complémentaires. La première est une chronologie des annonces, qui permet de suivre l’évolution des événements majeurs liés à l’IA en mettant en avant les dates clés, les sources et les titres des publications. La deuxième visualisation présente une répartition des thématiques par mot-clé sous forme de diagramme en camembert, mettant en lumière les sujets les plus fréquemment abordés, tels que l’éthique, la robotique ou la santé. La troisième partie consiste en un tableau répertoriant le top 100 des URL les plus citées, accompagné de leurs statistiques de publication. Enfin, la dernière visualisation est une carte des sources classées par éditeur ou organisation, permettant d’identifier les acteurs qui publient le plus d’actualités sur l’IA. Grâce à ces éléments, le tableau de bord propose une exploration à la fois intuitive, informative et visuellement structurée des tendances actuelles en intelligence artificielle.

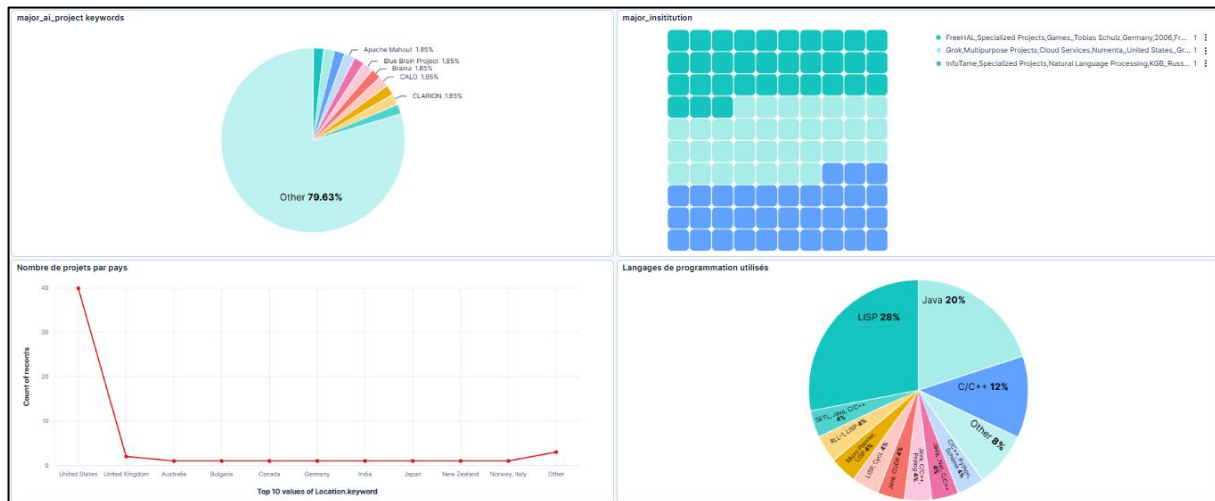


Figure 16: Tableau de bord Kibana : Analyse globale des projets en intelligence artificielle

Ce tableau de bord Kibana fournit une analyse synthétique et interactive des projets en intelligence artificielle, articulée autour de quatre visualisations principales. Un **diagramme circulaire des mots-clés** révèle les termes les plus récurrents, offrant un aperçu des thématiques dominantes et des technologies phares dans le domaine. Un **histogramme des institutions** illustre la répartition des projets par acteur, mettant en lumière les établissements les plus engagés dans la recherche et le développement en IA. La **carte des projets par pays** propose une perspective géographique, visualisant la concentration des initiatives à l'échelle mondiale. Enfin, un **diagramme circulaire des langages de programmation** expose les préférences technologiques des développeurs, clarifiant les outils les plus utilisés pour concrétiser ces projets. Ensemble, ces visualisations permettent une compréhension immédiate et structurée des tendances, acteurs et technologies façonnant l'écosystème de l'intelligence artificielle.

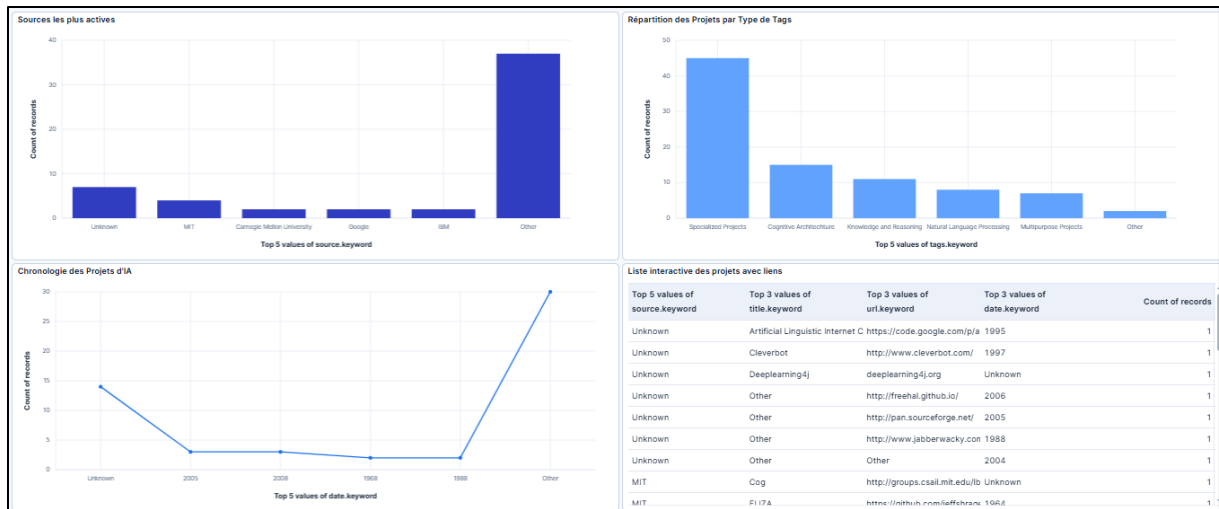


Figure 17: Tableau de bord Kibana : Exploration des articles Hacker News sur l'intelligence artificielle

Ce tableau de bord interactif offre une vue d'ensemble claire et synthétique des articles Hacker News en lien avec l'intelligence artificielle et la technologie. Il se compose de quatre visualisations principales : un diagramme en anneau représentant la répartition des tags (comme "AI", "tech") en pourcentage, permettant d'identifier rapidement les thématiques dominantes ; un graphique circulaire montrant la répartition des titres afin de visualiser les articles les plus fréquemment abordés ; un tableau cliquable listant les titres avec leurs liens directs, facilitant l'exploration et la lecture des contenus ; et enfin, un graphique en aires illustrant la distribution chronologique des articles pour observer leur évolution dans le temps. Ensemble, ces visualisations permettent une analyse rapide et efficace des tendances, des sujets populaires et de la temporalité des publications.

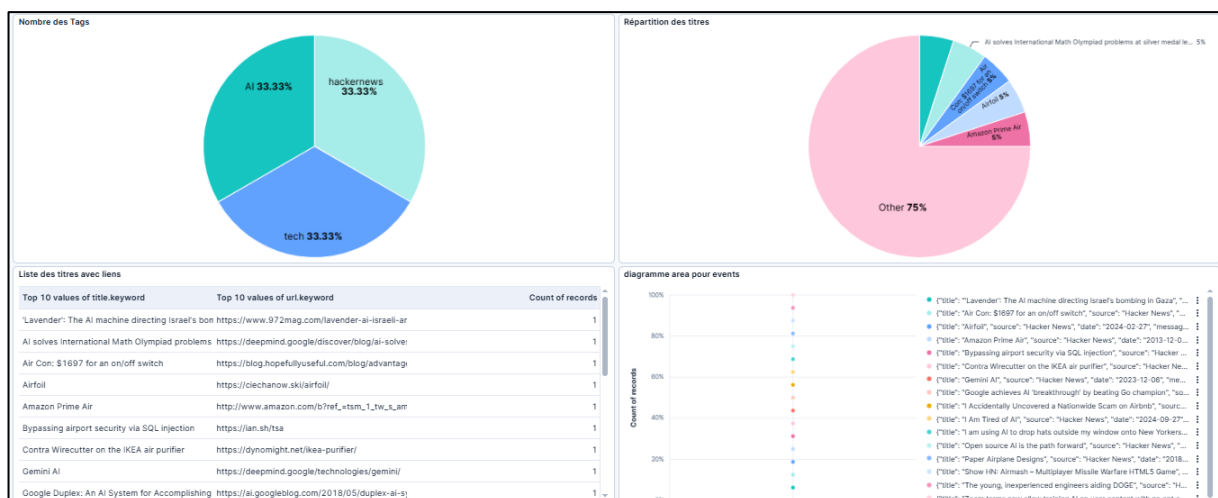


Figure 18: Tableau de bord Kibana : Analyse interactive des projets en intelligence artificielle

Ce tableau de bord interactif, conçu avec Kibana, offre une vue d'ensemble riche et structurée sur un ensemble de projets en intelligence artificielle. Il comprend quatre visualisations principales : un histogramme représentant la répartition des projets par date de création, un diagramme à barres illustrant les sources les plus actives, un nuage de mots-clés généré à partir des tags pour mettre en évidence les thématiques dominantes, et un tableau détaillé regroupant l'ensemble des informations descriptives sur chaque projet. Grâce à ce dashboard, il est possible d'identifier rapidement les tendances temporelles, les acteurs majeurs dans le domaine de l'IA, les domaines d'application les plus fréquents, ainsi que d'explorer facilement les caractéristiques spécifiques de chaque projet.

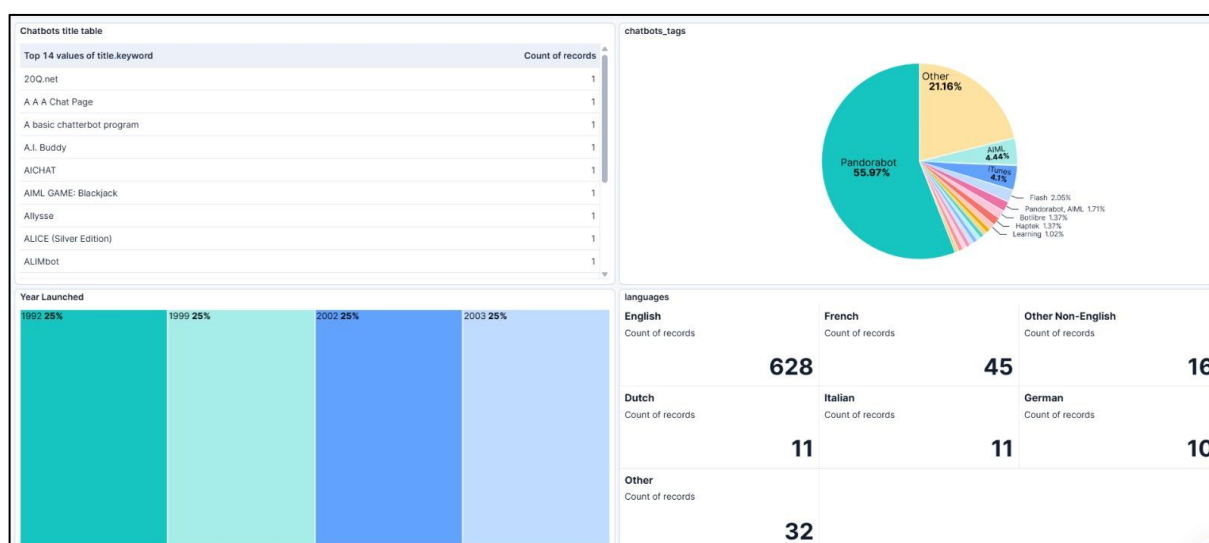


Figure 19: Tableau de bord Kibana : Exploration interactive des chatbots de la Chatterbot Collection

Ce tableau de bord propose une vue d'ensemble interactive d'une collection de chatbots issus de la *Chatterbot Collection*. Il s'ouvre sur un tableau récapitulatif affichant les titres des chatbots accompagnés du nombre de leurs occurrences, ce qui permet d'identifier les plus fréquemment mentionnés. Ensuite, un diagramme circulaire illustre la répartition des *tags* les plus courants, représentant les différentes catégories ou plateformes associées (comme *Pandorobot*, etc.). Un graphique en forme de *waffle chart* présente la répartition des années de lancement des chatbots, offrant ainsi une perspective temporelle sur leur évolution. Enfin, un tableau synthétique recense les langues supportées par les différents chatbots, avec le nombre de cas associés à chaque langue, fournissant une analyse linguistique pertinente de cette base de données.

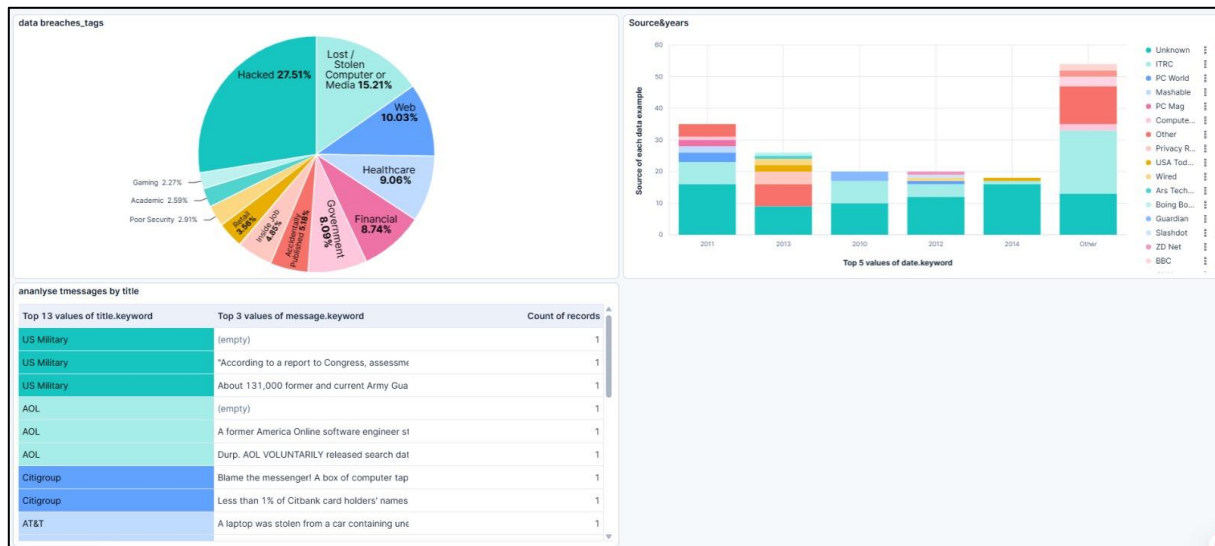


Figure 20: Tableau de bord Kibana : Analyse interactive des incidents de violation de données

Ce tableau de bord interactif présente une analyse synthétique de plusieurs incidents de violation de données. Il comprend trois visualisations principales. La première est un diagramme circulaire illustrant la répartition des tags associés aux incidents (comme *Hacked*, *Lost / Stolen Computer or Media*, *Accidentally Published*, etc.), permettant d'identifier les types de violations les plus fréquents. La deuxième visualisation est un diagramme à barres qui affiche, pour chaque année, les différentes sources ayant rapporté ces incidents, offrant une perspective temporelle et médiatique sur la couverture des fuites de données. Enfin, un tableau dynamique recense les messages associés à chaque incident, classés par titre, fournissant un aperçu détaillé du contenu et du contexte de chaque cas. Ce tableau de bord facilite ainsi la compréhension globale des tendances en matière de sécurité des données.

Conclusion générale

Ce projet a marqué une étape significative dans notre exploration des systèmes distribués et des technologies modernes de gestion de données. En développant un moteur de recherche basé sur la stack ELK (Elasticsearch, Logstash, Kibana), nous avons relevé avec succès le défi de traiter des données hétérogènes tout en offrant une expérience utilisateur fluide et performante. L'interface web développée avec Flask a permis de rendre la solution accessible, tandis que les fonctionnalités avancées d'indexation et de recherche ont démontré l'efficacité d'une architecture distribuée face à des volumes importants de données.

Au-delà des résultats techniques, cette expérience nous a apporté des enseignements précieux. La gestion des différents formats de données et la configuration des pipelines d'ingestion nous ont confrontés à des défis stimulants, renforçant notre capacité à concevoir des solutions robustes et adaptables. Les limites rencontrées avec certaines API publiques nous ont également sensibilisés à l'importance de la qualité et de la disponibilité des données dans un contexte réel.

En perspective, ce projet ouvre la voie à de nombreuses améliorations passionnantes. L'intégration de l'intelligence artificielle, notamment via des modèles NLP, pourrait transformer l'expérience de recherche en la rendant plus intuitive et pertinente. Par ailleurs, le passage à une infrastructure cloud avec Docker et Kubernetes permettrait d'optimiser la scalabilité et la maintenance du système. Ces évolutions positionnent notre solution comme une base solide pour des applications futures plus ambitieuses.

Enfin, ce travail a renforcé notre compréhension des enjeux liés aux systèmes distribués, tout en mettant en lumière l'importance d'une approche structurée et résiliente. Les compétences acquises, tant sur le plan technique que méthodologique, constituent un atout majeur pour aborder des projets complexes dans le domaine du Big Data et de l'IA. Ce moteur de recherche n'est donc pas seulement une réalisation concrète, mais aussi un tremplin vers de nouvelles opportunités d'innovation.

Références

Documentation technique utilisée

- Documentation Elasticsearch : <https://www.elastic.co/guide/en/elasticsearch/>
- Documentation Logstash : <https://www.elastic.co/guide/en/logstash/>
- Documentation Kibana : <https://www.elastic.co/guide/en/kibana/>
- Documentation Flask : <https://flask.palletsprojects.com/>
- Tutoriels sur Elasticsearch & Logstash : YouTube, OpenClassrooms

Sources de données et APIs

- GitHub – Repositories contenant des jeux de données JSON sur l’IA et l’informatique (ex. : Data-Breaches, Major A.I. Projects.csv.csv) : [datasets/Artificial-Intelligence at master · ali-ce/datasets](#) /// [datasets/Biggest-Data-Breaches at master · ali-ce/datasets](#)
- Kaggle – Jeux de données CSV sur des projets en IA/IT : <https://www.kaggle.com/>
- Hacker News API (via Algolia) – Pour extraire des articles et discussions techniques sur l’intelligence artificielle : <https://hn.algolia.com/api>