RÉPUBLIQUE DU SÉNÉGAL

Un peuple - un but - une foi

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR, DE LA RECHERCHE ET DE L'INNOVATON

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

Foo Nekk Foofu La

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

POLE SCIENCE, TECHNOLOGIES ET NUMERIQUE
MASTER : BIG DATA ANALYTICS & INTELLIGENCE ARTIFICIELLE

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

Sujet :

Etude et Conception d'un crawler et scraper intelligents large spectre

¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤¤

Présenté par :

M. PAPA SEYDOU WANE

M. ISMAÏLA GNING

Sous la direction de :
Dr Edouard Ngor SARR
Sous la supervision de :
Prof Ousmane SALL

Membres du jury :
Président : **Pr Alassane DIOP**
Examinateur 1 : **Haby DIALLO**
Examinateur 2 : **Maurice Djibril FAYE**
Examinateur 3 : **Gorgoumack SAMBE**

Année universitaire : 2022 – 2023

Soutenu publiquement **le 22 mais 2025**

Dédicace

Ce mémoire est dédié à nos familles, pour leur soutien inconditionnel et leurs encouragements tout au long de notre parcours académique.

Remerciements

Nous souhaitons exprimer notre profonde gratitude à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce mémoire.

- À notre directeur de mémoire, Dr Edouard Ngor SARR, pour son encadrement, ses conseils avisés et sa patience tout au long de ce travail. Son expertise et ses orientations méthodologiques ont été précieuses pour mener à bien ce projet.
- À notre superviseur de mémoire, Professeur Ousmane SALL, Vice-Recteur, pour son engagement dans la formation et l'accompagnement des étudiants nous a été d'une grande aide.
- Aux membres du jury, pour avoir accepté d'évaluer ce travail et pour leurs remarques constructives qui contribueront à son amélioration.
- À l'ensemble des professeurs de la filière, dont l'enseignement et les conseils ont nourri notre réflexion et nous ont guidés tout au long de notre parcours universitaire.
- Au corps professoral de l'Université Numérique Cheikh Hamidou Kane, pour leur disponibilité et leur engagement dans l'excellence académique.
- Au personnel administratif et technique du Campus ATOS de l'UNCHK, pour leur soutien logistique et leur accompagnement.
- À nos familles, pour leur amour, leur patience et leur soutien moral sans faille. Leur présence a été une source de motivation dans les moments de doute et de fatigue.
- À nos amis et camarades de promotion, pour les échanges, l'entraide et les moments de partage qui ont rendu cette aventure académique plus enrichissante et agréable.
- Enfin, à tous ceux qui ont contribué, de près ou de loin, à la réussite de ce projet, nous leur adressons nos sincères remerciements.

Nous espérons que ce travail sera une contribution utile à la communauté académique et professionnelle, et qu'il pourra servir de point de départ pour de nouvelles recherches et applications dans le domaine du crawling, du scraping et de l'intelligence artificielle appliquée à l'extraction de données.

Résumé

Dans ce mémoire, nous abordons la problématique liée à l'exploration et l'extraction intelligentes des données à partir des corpus web. Nous y proposons une système combinant du Web Crawling, du Web Scraping et de l'intelligence Artificielle afin d'automatiser l'extraction de données ciblées. L'objectif générale visé ici est de permettre un accès facile à des contenus ciblés. En d'autres termes, nous cherchons à (i) faciliter l'exploration des sites web en récupérant automatiquement leurs contenus pertinents, (ii) à Filtrer et classer les informations en fonction de leur lien avec le sujet traité grâce aux techniques de NLP et (iii) à offrir une base de données centralisée et exportable sous divers formats (Excel, JSON) pour une exploitation ultérieure. Ce travail est divisé en quatre chapitres. Le premier chapitre établit le cadre général et la problématique, le second présente un état de l'art des techniques existantes et leur positionnement, le troisième détaille le cahier des charges et le choix des technologies, et enfin, le quatrième chapitre expose la solution développée, ainsi que les tests effectués pour valider son efficacité. Ce travail s'inscrit ainsi dans une dynamique de valorisation du patrimoine historique du Sénégal en exploitant les avancées technologiques en matière de crawling, scraping et intelligence artificielle.

Abstract

In this thesis, we address the challenges related to the exploration and intelligent extraction of data from web corpora. We propose a system combining web crawling, web scraping, and artificial intelligence to automate the extraction of targeted data. The overarching objective is to enable easy access to curated content. Specifically, we aim to: (i) streamline the exploration of websites by automatically retrieving their relevant content, (ii) filter and classify information based on its relevance to the subject matter using NLP techniques, and (iii) provide a centralized, exportable database in various formats (Excel, JSON) for further analysis. This work is structured into four chapters. The first chapter establishes the general framework and outlines the core challenges. The second reviews the state of the art in existing techniques and their applications. The third details the project specifications and technology choices, while the fourth presents the developed solution and the tests conducted to validate its effectiveness.
This research aligns with efforts to promote Senegal's historical heritage by leveraging advancements in web crawling, scraping, and artificial intelligence technologies.

Sommaire

Table des matières

Liste des figures

Listes des tableaux

Sigles et abréviation

- **RI** : Recherche d'Information
- **IA** : Intelligence Artificielle
- **NLP :** Traitement du Langage Naturel
- **IEEE :** Institute of Electrical and Electronics Engineers
- **TF-IDF :** Term Frequency-Inverse Document Frequency
- **HTML :** HyperText Markup Language
- **XHTML :** Extensible HyperText Markup Language
- **XML :** Extensible Markup Language
- **DOM :** Document Object Model
- **RGPD** : Règlement Général sur la Protection des Données
- **API** : Application Programming Interface
- **GET** : Une méthode HTTP pour récupérer des données à partir d'un serveur.
- **POST** : Une méthode HTTP utilisée pour envoyer des données au serveur pour traitement.
- **URL** : Uniform Resource Locator
- **HTTP** : Hypertext Transfer Protocol
- **VPN** : Virtual Private Network
- **IP** : Internet Protocol.
- **ANN** : Artificial Neural Network
- **LSTM** : Long Short-Term Memory
- **AWS** : Amazon Web Services
- **RL** : Reinforcement Learning

Introduction Générale

Dans un contexte de croissance exponentielle de la quantité de données disponibles en ligne, il devient crucial d'extraire ces informations de manière efficace et intelligente. Malgré leur efficacité, les moteurs de recherche classiques présentent des limites face à la quantité et à la complexité des données actuelles. Les algorithmes de recherche classiques rencontrent des difficultés à fournir des résultats pertinents et actualisés, ce qui entraîne des pertes de temps et des inefficacités pour les utilisateurs et les organisations. C'est là qu'intervient l'intelligence artificielle (IA), proposant des solutions novatrices pour créer des robots intelligents capables de naviguer et d'analyser le web de manière autonome et performante. Ces robots de recherche intelligents exploitent des méthodes comme l'apprentissage automatique, le traitement du langage naturel et les réseaux neuronaux afin de saisir le contenu des pages web, repérer les informations pertinentes et les extraire de manière optimale.

Les études récentes mettent en évidence l'efficacité de ces nouvelles méthodes. À titre d'exemple, il est démontré comment l'apprentissage automatique peut apporter une amélioration considérable à la précision des robots de recherche en ligne. Leur recherche met l'accent sur l'utilisation de modèles de classification afin de donner la priorité aux pages à explorer, ce qui permet de diminuer le temps et les ressources nécessaires pour recueillir des données de grande qualité **[1]**. L'utilisation des réseaux neuronaux dans la création de crawlers intelligents permet de saisir non seulement le contenu textuel, mais aussi des éléments contextuels et sémantiques, offrant ainsi une collecte d'informations plus pertinente et précise. Leur étude souligne également l'importance de l'auto-apprentissage des robots, leur permettant de s'ajuster en temps réel aux évolutions dynamiques de l'architecture du web **[2].**

Ainsi, l'intégration de l'IA dans la conception des crawlers web ouvre de nouvelles perspectives, rendant les processus de collecte de données en ligne plus intelligents, rapides et précis. Cependant, malgré ces avancées, les robots web classiques rencontrent encore des contraintes importantes dues à l'augmentation massive des informations disponibles en ligne et à la variété des formats de données.

Les principales difficultés liées à la collecte de données sur le web sont :

- La capacité à repérer et extraire des informations pertinentes parmi des milliards de pages web.
- La gestion des évolutions dynamiques du contenu en temps réel.
- Le traitement efficace de données non structurées ou semi-structurées.

Ces limitations soulèvent une question centrale : **Comment l'intelligence artificielle peut-elle améliorer la fouille et la collecte de données dans le web ?**

En d'autres termes, quelles approches basées sur l'IA permettent de concevoir des crawlers et scrapers intelligents capables de surmonter ces défis ?

Pour répondre à cette problématique, ce mémoire propose la conception d'un crawler et scraper intelligents large spectre, capables de :

- Identifier et extraire automatiquement des informations pertinentes sur le web;
- S'adapter en temps réel aux changements dynamiques des pages web;
- Exploiter des algorithmes d'apprentissage automatique pour optimiser les ressources et prioriser les pages web à explorer;
- Intégrer des technologies avancées comme le traitement du langage naturel et les réseaux neuronaux pour une meilleure analyse contextuelle et sémantique des données collectées.

L'objectif principal de ce mémoire est d'identifier et de proposer des solutions d'utilisation de l'intelligence artificielle pour surmonter les obstacles liés au web crawling et scraping. Plus spécifiquement, il s'agit de :

- Étudier les approches actuelles et identifier leurs limitations;
- Concevoir un prototype intégrant des technologies de l'IA;
- Évaluer les performances du prototype et proposer des pistes d'amélioration.

Cette étude présente plusieurs intérêts :

- Académiques : pour enrichir les recherches sur l'utilisation de l'IA dans le web crawling et scraping;
- Pratiques : pour fournir un outil performant et adaptable aux besoins des chercheurs et des entreprises pour extraire des données de qualité à grande échelle.

Pour répondre aux questions soulevées, une approche mixte sera adoptée :

- Qualitative : pour l'analyse des travaux existants pour identifier les opportunités d'innovation.
- Quantitative : pour le développement et expérimentation d'un prototype pour évaluer ses performances.

L'approche mixte est justifiée par la nécessité d'allier une compréhension approfondie du contexte théorique à une évaluation empirique des solutions proposées.

Nos travaux sont divisés en quatre grands chapitres qui sont :

1. **Chapitre 1 : Généralités et problématique**

Introduction aux concepts de web crawling et scraping, ainsi qu'aux défis qui y sont associés.

0. **Chapitre 2 : Crawling et scraping intelligents**

   Étude des approches intelligentes, des outils et des technologies actuelles, ainsi que des obstacles rencontrés.

0. **Chapitre 3 : État de l'art**

   Exploration des travaux scientifiques sur l'intégration de l'IA dans le web crawling et scraping, et positionnement critique.

0. **Chapitre 4 : Proposition et expérimentation**

   Présentation de notre solution, développement d'un prototype, et évaluation des performances avec des données réelles.

Chapitres 1 : Généralité et problématique

1.1. Généralité

Depuis sa création au début des années 1990, le World Wide Web**,** communément appelé Web[1]**,** a connu une croissance exponentielle. Il existait qu'environ une centaine de sites en 1993, sa taille estimée dépasse aujourd'hui les 1.98 milliards de sites web **[3].** Effet direct de cette croissance, les données qu'il contient sont progressivement devenues plus vastes et extrêmement intéressantes pour les entreprises. Un nouveau défi est alors apparu : elles doivent apprendre à recueillir, comprendre et exploiter les informations issues de cette source quasi inépuisable et sans cesse renouvelée. La multiplication des comparateurs pour tous les types de produits pour les assuranceurs, les banques, les produits culturels, etc. qui centralisent les données collectées de différents sites. On peut également imaginer une enseigne de supermarché étudiant la concurrence de façon automatique pour ajuster ses prix **[4].** C'est dans ce cadre que s'inscrivent les concepts de crawling et scraping, outils majeurs de la collecte de données sur le Web. Nous étudierons les types de web crawling et de web scraping. Par la suite, nous mettrons en relief les exemples de web scraping et de web crawling par le biais d'un cas d'utilisation. Et pour terminer, nous ferons le point sur une étude comparative entre le web scraping et le web crawling.

1.1.1. La recherche d'information (RI)

1.1.1.1. La Recherche d'Information

La recherche d'information (RI) est devenue un processus crucial dans un monde où les données numériques croissent de manière exponentielle. Elle se définit comme le processus de récupération et d'organisation d'informations pertinentes à partir de grandes bases de données ou d'Internet. Cela peut inclure des outils comme Google[2], des bases scientifiques telles que Hal Science[3], ou même des systèmes de gestion des connaissances en entreprise. L'objectif principal de la RI est de fournir des résultats précis et pertinents tout en optimisant le temps et les ressources nécessaires. Les applications de la RI sont vastes : des chercheurs accédant à des articles académiques, des médecins utilisant des plateformes spécialisées, jusqu'aux utilisateurs quotidiens cherchant des informations sur des produits ou des services. Grâce aux avancées en

[1] Ensemble des données reliées par des liens hypertextes, sur Internet.
[2] Un moteur de recherche sur Internet gratuit et libre d'accès sur le World Wide Web,
[3] Une ressource gratuite soutenant la recherche scientifique.

intelligence artificielle et au traitement du langage naturel, les capacités de la RI sont optimisées pour maximiser leur efficacité.

1.1.1.2. Les étapes de recherche d'information

a) Formulation de la requête

La première étape de la recherche d'information est de définir clairement les besoins de l'utilisateur. Cette phase repose sur la formulation de mots-clés spécifiques ou de requêtes en langage naturel. Une bonne formulation est cruciale pour garantir la qualité des résultats obtenus. Par exemple, l'utilisation de mots-clés précis et de connecteurs logiques peut affiner les résultats et réduire les informations inutiles (ou bruit). Ces techniques, comme l'analyse des intentions utilisateur et le traitement du langage naturel, permettent également d'interpréter des requêtes complexes ou mal formulées, rendant le processus plus efficace. Cette étape est essentielle pour orienter les systèmes vers les contenus les plus pertinents.

b) Interrogation des systèmes

Une fois la requête clairement définie, elle peut être soumise à des systèmes de recherche comme les moteurs de recherche web tels que Google et Bing. Ces systèmes utilisent des indices sophistiqués pour analyser et sélectionner les documents pertinents. Par exemple, Google applique des algorithmes comme PageRank pour attribuer une valeur plus élevée à certains résultats, tandis que les bases de données académiques s'appuient principalement sur des métadonnées et des ontologies pour classer les publications. Cette étape est essentielle pour simplifier la navigation à travers de vastes ensembles de données.

c) Traitement des données

Les données collectées sont ensuite classées et filtrées en fonction de leur pertinence pour l'utilisateur, en utilisant des algorithmes comme TF-IDF (Term Frequency-Inverse Document Frequency) ou des modèles probabilistes pour évaluer la pertinence des documents. Ce traitement permet de réduire le bruit tout en mettant en évidence les informations les plus importantes. Dans les systèmes avancés, des techniques comme le tri automatique par catégorie ou le regroupement thématique sont également appliquées, facilitant l'analyse des résultats. Ainsi, seules les données pertinentes sont présentées à l'utilisateur, optimisant ainsi la phase suivante.

d) Evaluation des résultats

L'utilisateur évalue ensuite la pertinence des résultats en fonction de ses besoins initiaux. Cette étape est intrinsèquement subjective, car elle repose sur la capacité de l'utilisateur à percevoir si les réponses satisfont ses attentes. Les interfaces modernes facilitent cette évaluation en

offrant des extraits textuels, des mots-clés surlignés, et des outils interactifs de tri. Par ailleurs, ces évaluations permettent aux systèmes de recherche d'ajuster leurs algorithmes grâce à des mécanismes de rétroaction, afin de mieux comprendre les préférences de l'utilisateur **[5].** Cette interaction directe améliore considérablement la satisfaction et l'efficacité de la recherche.

e) Rétroaction

La rétroaction constitue une boucle essentielle dans le processus de recherche d'information, car elle permet à l'utilisateur d'ajuster sa requête pour obtenir des résultats plus pertinents. Par exemple, cela peut inclure l'ajout ou la suppression de mots-clés, ou l'application de filtres comme la date de publication ou la source. Les systèmes modernes, notamment ceux reposant sur l'intelligence artificielle, utilisent cette rétroaction pour apprendre et améliorer continuellement leurs algorithmes, rendant ainsi les recherches futures plus précises et rapides. Cette étape dynamique garantit une amélioration constante du processus de recherche **[6].**

1.1.1.3. Modèles théoriques de la recherche d'information

a) Le modèle Booléen

Le modèle booléen utilise des opérateurs logiques tels qu'AND, OR, et NOT pour filtrer les résultats en fonction de la présence ou de l'absence de certains termes. Ce modèle est apprécié pour sa simplicité, permettant de formuler des requêtes claires, par exemple, COVID AND Vaccin NOT Controverse. Mais aussi il faut noter qu'il présente des limites en matière de flexibilité, car il ne classe pas les résultats par pertinence et n'inclut pas les documents partiellement correspondants. Ainsi, ce modèle peut soit générer un grand nombre de résultats non pertinents, soit exclure certains résultats utiles **[7] [9].**

b) Modèle vectoriel

Le modèle vectoriel attribue des poids aux termes d'une requête et aux documents, ce qui permet de mesurer leur similarité. Chaque terme est représenté par un vecteur, et la pertinence est calculée en fonction de la distance entre le vecteur du document et celui de la requête. Contrairement au modèle booléen, il permet de classer les résultats, en donnant une priorité aux documents les plus proches de la requête en termes de contenu. Ce modèle est particulièrement adapté pour les recherches où les mots-clés sont approximatifs ou incomplets **[9] [6].**

c) Modèle probabiliste

Le modèle probabiliste évalue la probabilité qu'un document soit pertinent pour une requête donnée. Cette probabilité est calculée à partir de données historiques ou statistiques, telles que la fréquence d'apparition des termes dans les documents jugés pertinents. Ce modèle offre un classement basé sur ces probabilités, maximisant ainsi les chances de trouver des informations

utiles en haut des résultats. Bien que puissant, il dépend fortement de la disponibilité de données d'entraînement et de l'ajustement précis des paramètres probabilistes **[8] [9].**

d) Modèle basée sur le langage

Le modèle basé sur le langage interprète les requêtes et les documents comme des échantillons d'un même espace linguistique. Il génère des scores de pertinence en évaluant la probabilité qu'un document ait produit la requête donnée. Ce modèle dépasse les simples correspondances de mots-clés en intégrant une compréhension contextuelle et sémantique. Il peut reconnaître des synonymes ou des expressions équivalentes, ce qui améliore considérablement la pertinence des résultats. Il est largement utilisé dans les systèmes modernes de recherche grâce à sa flexibilité et son efficacité **[10] [11].**

1.1.1.4. Technologies et Outils de la Recherche d'Information

A) Langage de balisage

Les langages de balisage sont des outils fondamentaux pour structurer, organiser et présenter des données dans divers contextes numériques. Ils permettent de représenter les informations à l'aide de balises qui définissent leur structure, leur signification et leurs relations. Les langages les plus couramment utilisés incluent HTML, XHTML et XML, chacun étant adapté à des besoins spécifiques.

- **HTML ET XHTML**

Les HTML est le langage standard pour structurer et afficher les pages Web. Il sert à baliser les contenus textuels et multimédias tels que les titres, paragraphes, images et liens hypertextes. HTML joue un rôle central dans l'architecture du Web, facilitant la navigation et l'interaction des utilisateurs. Des moteurs de recherche comme Google indexent les pages Web en analysant leur structure HTML.

XHTML est une version plus rigoureuse de HTML, basée sur les règles de XML. Il garantit une meilleure compatibilité entre les systèmes grâce à des normes plus strictes, notamment la fermeture obligatoire des balises, une hiérarchie stricte des éléments imbriqués et une syntaxe uniforme pour les développeurs.

Ces deux langages favorisent non seulement l'affichage des informations, mais aussi leur indexation par des moteurs de recherche, augmentant leur visibilité et leur accessibilité. Par exemple, une page structurée en XHTML est plus facilement analysée par des systèmes automatisés, améliorant son classement dans les résultats de recherche **[12].**

```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <div>
        <p>je suis un paragraphe</p>
        <img src="" alt="une  image">
        <table>je suis un tableau</table>
    </div>
</body>
</html>
```

Figure 1: Structure d'une page HTML

- **XML**

XML est un langage universel conçu pour structurer les données de manière hiérarchique. Contrairement à HTML et XHTML, qui se concentrent sur l'affichage, XML est principalement utilisé pour stocker et transporter des données de manière indépendante des plateformes ou applications.

Les principaux avantages de XML incluent sa flexibilité, parce qu'il permet de créer des balises personnalisées pour répondre à des besoins spécifiques. Dans un fichier XML pour une bibliothèque, des balises comme <auteur>, <titre>, et <année> peuvent être utilisées pour organiser les informations. Son interopérabilité, grâce à sa structure universelle, XML peut être utilisé dans divers systèmes tels que les bases de données, applications Web, ou logiciels de gestion. Son organisation hiérarchique, ces données XML sont structurées sous forme d'arbre, facilitant leur navigation et manipulation. Par exemple, une application peut récupérer uniquement les parties spécifiques d'un document XML, comme tous les livres publiés après une certaine date.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<catalogue xml:lang="fr" xml:space="preserve" xml:base="http://www.example.com/livres/">
    <!-- Ceci est un commentaire : catalogue de livres -->
    <livre id="1" genre="Roman">
        <titre>Les Misérables</titre>
        <auteur>Victor Hugo</auteur>
        <publication>1862</publication>
        <description><![CDATA[
            Un classique de la littérature française décrivant les injustices sociales.
        ]]></description>
    </livre>
    <livre id="2" genre="Science-Fiction">
        <titre>Dune</titre>
        <auteur>Frank Herbert</auteur>
        <publication>1965</publication>
        <description>Un roman épique se déroulant dans un univers désertique.</description>
    </livre>
</catalogue>
```

Figure 2: Structure d'une page XML

B) DOM ET XPATH

- **DOM**

Le Document Object Model est une interface standardisée qui représente la structure d'un document XML ou HTML sous forme d'un arbre de nœuds. Chaque nœud représente un élément, un attribut ou un contenu textuel, pouvant être modifié dynamiquement par des programmes. Dans une application Web, le DOM permet de changer la mise en page ou d'ajouter des interactions sans recharger la page entière. Il est particulièrement utile pour les documents XML, car il fournit une base pour explorer, insérer ou supprimer des éléments selon les besoins. Un avantage clé du DOM est son indépendance vis-à-vis des plateformes et des langages, ce qui le rend compatible avec divers environnements. Toutefois, pour les très grands documents, sa consommation mémoire peut être élevée, car il charge le document entier en mémoire pour le manipuler **[12] [14].**

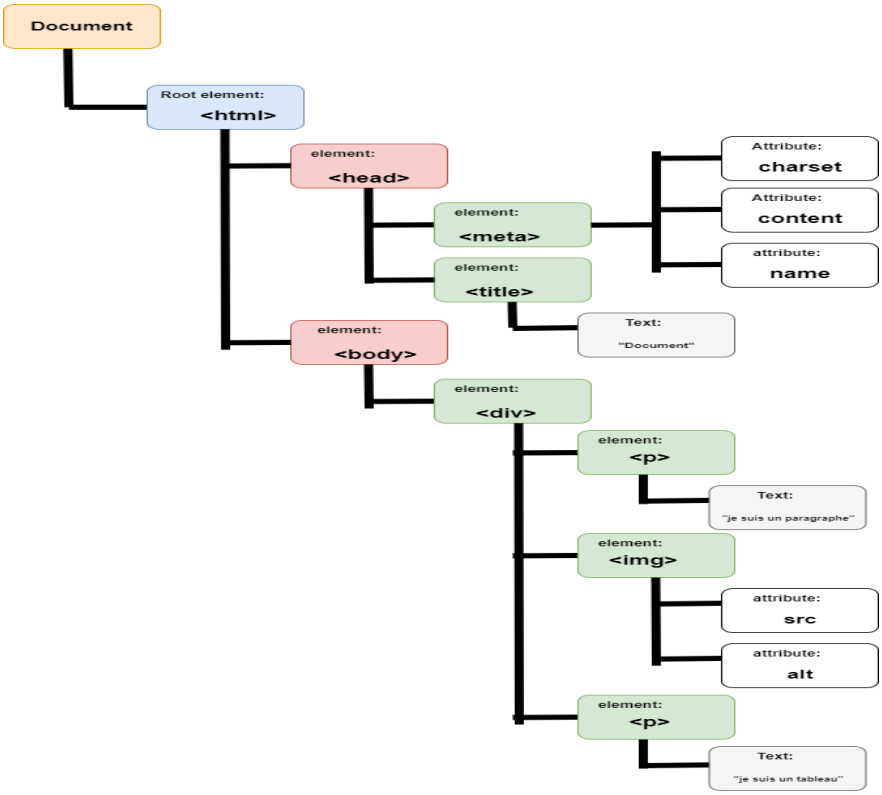


Figure 3: Représentation DOM

- **XPATH**

XPath est un langage de requête conçu pour interroger et extraire des données des documents XML ou XHTML en parcourant leur arborescence DOM. Il permet de cibler des éléments ou des attributs selon des critères spécifiques. Par exemple, une expression XPath comme //livre/auteur permet de récupérer tous les auteurs dans une base de données XML contenant des livres. L'efficacité d'XPath réside dans sa capacité à naviguer dans des structures complexes et

à isoler les données pertinentes sans traitement manuel. Très utilisé dans des domaines comme les flux RSS, les services Web et les bases de données XML, XPath est souvent intégré à des technologies comme XSLT pour transformer des documents ou XQuery pour interroger de vastes ensembles de données **[13] [4].**

- **Intégration du DOM ET XPATH**

Le DOM et XPath sont souvent combinés pour maximiser l'efficacité de la gestion et de l'interrogation des documents XML ou XHTML. Le DOM fournit une structure arborescente permettant de naviguer et manipuler les éléments, tandis qu'XPath offre une méthode précise pour localiser et extraire des données spécifiques dans cette structure. Par exemple, dans une application de gestion de bibliothèque, le DOM pourrait représenter la base de données des livres, tandis qu'XPath serait utilisé pour rechercher des ouvrages par auteur ou par année de publication. Cette combinaison permet de traiter rapidement des informations complexes tout en minimisant les efforts de développement **[12] [15].**

C) Algorithme de recherche d'information

- **PageRank**

Le PageRank est un algorithme inventé par les fondateurs de Google, Larry Page et Sergey Brin, qui attribue une valeur aux pages web en fonction de leur importance. L'algorithme considère chaque page comme un nœud dans un graphe et les liens hypertextes entre les pages comme des arêtes. Le PageRank d'une page est proportionnel au nombre de liens entrants provenant de pages bien notées. Cette approche, basée sur la théorie des graphes et les chaînes de Markov, a radicalement transformé la manière dont les moteurs de recherche classent les résultats. Bien qu'initialement conçu pour le web, PageRank est également applicable dans d'autres domaines, comme l'analyse des réseaux sociaux et bibliométriques **[12] [13].**

- **TF-IDF (Term Frequency-Inverse Document Frequency)**

TF-IDF est une technique permettant de représenter numériquement l'importance d'un terme dans un document par rapport à un ensemble de documents. Elle repose sur deux mesures :

- TF qui est la fréquence d'apparition d'un terme dans un document.
- IDF qui quant à lui est la mesure inverse de la fréquence d'apparition de ce terme dans le corpus.

Cette combinaison attribue un poids élevé aux termes fréquents dans un document donné mais rares dans l'ensemble du corpus. Cette méthode est largement utilisée dans les systèmes de

recherche d'informations où le contenu textuel est analysé pour évaluer la pertinence des documents **[13] [16].**

- **Bases de données spécialisées**

Les bases de données comme PubMed dans le domaine médical et IEEE Xplore pour la recherche technologique, offrent des systèmes de recherche adaptés à des contextes spécifiques. Elles permettent des requêtes avancées avec des filtres précis, comme la date de publication, les auteurs ou les mots-clés thématiques. Contrairement aux moteurs de recherche généraux, ces bases de données sont souvent payantes et ciblent des utilisateurs professionnels ou académiques **[15] [16].**

1.1.1.5. Enjeux et Defis

A) Big Data

Avec l'augmentation exponentielle des données numériques générées chaque jour, les systèmes de recherche d'information doivent relever des défis liés à la scalabilité, la rapidité et la précision. Les ensembles massifs de données, souvent non structurées, nécessitent des outils capables de traiter des volumes importants tout en identifiant rapidement les informations pertinentes. Par exemple, dans le domaine des sciences biologiques, des bases de données comme celles utilisées pour le séquençage génomique atteignent plusieurs pétaoctets, posant des problèmes de stockage, de traitement et de partage des données [17] [18].

B) Multilinguisme

Dans un monde globalisé, l'information est produite dans une grande variété de langues comme le wolof, ce qui rend crucial le besoin de systèmes capables de gérer et de traduire efficacement ces données. Les approches multilingues et les systèmes de récupération d'information inter linguistiques permettent de dépasser ces obstacles linguistiques. Par exemple, l'utilisation du wolof combiner avec le français et le langage urbain, illustre comment l'intégration de données multilingues peut améliorer l'accessibilité à l'information **[19].** Toutefois, ces systèmes rencontrent des difficultés liées à la qualité des traductions automatiques et à l'ambiguïté linguistique **[20].**

C) Pertinence et Bruit

Avec l'augmentation des données personnelles dans les systèmes de recherche, la confidentialité et l'éthique sont devenues des préoccupations centrales. Les utilisateurs souhaitent accéder à des informations pertinentes sans compromettre leur vie privée. Au Sénégal la **loi n°2008-12 du 25 janvier 2008 portant sur la protection des données à caractère personnel** impose des restrictions strictes sur la collecte et l'utilisation des données

personnelles **[24].** Par ailleurs, des mécanismes tels que l'anonymisation des données et le contrôle des autorisations sont indispensables pour assurer la conformité légale et préserver la confiance des utilisateurs **[25].**

D) Ambiguïté Sémantique

Les mots peuvent porter plusieurs significations selon le contexte, ce qui complique la compréhension et la récupération d'informations. Par exemple, le terme "banc" peut désigner un meuble ou une institution financière. Les modèles de recherche doivent intégrer des techniques d'analyse sémantique afin de désambiguïser ces termes en fonction de leur usage dans un contexte donné. Des projets comme l'utilisation de graphes sémantiques illustre comment les relations entre les concepts peuvent être exploitées pour résoudre cette ambiguïté et optimiser les résultats des recherches **[23].**

E) Confidentialité et Ethique

Avec l'augmentation des données personnelles dans les systèmes de recherche, la confidentialité et l'éthique occupent une place prioritaire **[22].** La plupart des utilisateurs souhaitent accéder à des informations pertinentes sans sacrifier leur vie privée **[21].**

1.1.1. Fouille et collecte de données web

La fouille et la collecte de données web englobent des méthodes et techniques sophistiquées visant à explorer, extraire et analyser les informations disponibles sur le web. Située à la croisée de l'informatique, des statistiques et de l'intelligence artificielle, cette discipline se divise en plusieurs branches : la fouille du contenu, qui se concentre sur l'analyse des données textuelles et multimédias, la fouille de la structure, qui examine les liens et connexions entre les pages, et la fouille de l'utilisation, qui s'intéresse aux comportements des utilisateurs. En parallèle, la collecte de données web se concentre sur le processus d'extraction automatique d'informations en ligne à l'aide d'outils dédiés.

1.1.1.1. Fouille de donnée web

A) Fouille du contenu web

La fouille du contenu se concentre sur l'analyse des textes, images, vidéo et autres types de contenus multimédias présents sur les pages web. Cette approche permet d'extraire des informations pertinentes grâce à des techniques telles que le traitement du langage naturel, l'analyse d'images et les algorithmes de classification. Par exemple, un moteur de recherche peut identifier des mots-clés ou générer des résumés de documents pour améliorer la pertinence des résultats [26].

Cependant, la diversité des formats de contenu que ça soit du PDF, des vidéos ou des contenus non structurés, ils représentent un défi technique important, en particulier pour la reconnaissance et la structuration des données.

B) Fouille de la structure du web

La fouille de la structure se concentre sur l'exploration des relations entre les pages web via leurs liens hypertexte. Cette approche utilise des graphes pour analyser la connectivité des pages, détecter des clusters et identifier des communautés. Des algorithmes comme PageRank, employé par Google, illustrent cette méthode en attribuant un score d'importance aux pages en fonction de leur degré de connectivité **[27] [5].**

Mais les modifications fréquentes dans la structure des sites web, l'ajout ou la suppression de liens compliquent cette analyse et nécessitent une mise à jour régulière des graphes.

C) Fouille de l'utilisation du web

Cette catégorie se concentre sur l'analyse des comportements des utilisateurs à partir de données collectées via les journaux de navigation, cookies ou clics. Elle offre une compréhension approfondie des habitudes et préférences des internautes, permettant ainsi de personnaliser les services et d'optimiser les plateformes, notamment dans le secteur du e-commerce **[28].**

Le respect de la vie privée est un défi crucial, particulièrement lorsque les données collectées incluent des informations personnelles sensibles **[29].**

1.1.1.2. Collecte de données web

La collecte de données web désigne le processus visant à récupérer des informations disponibles en ligne pour les analyser ou les intégrer dans des systèmes d'information. Contrairement à la fouille de données, qui analyse des données déjà collectées, la collecte de données se concentre sur le recueil initial des informations. Ce processus inclut diverses méthodes et outils conçus pour extraire, structurer et organiser des données provenant de sources multiples, telles que les sites web, les bases de données en ligne ou les API **[30].**

A) Applications de la collecte de données web

La collecte de données web est utilisée dans de nombreux secteurs pour atteindre des objectifs spécifiques :

- La surveillance des prix, les entreprises s'appuient sur des outils automatisés pour suivre l'évolution des prix sur les plateformes de commerce électronique. Cette pratique leur permet d'ajuster leur stratégie tarifaire en temps réel et de maintenir leur compétitivité sur le marché.

- La recherche scientifique et académique, la collecte de données en ligne est employée pour réunir des informations provenant de diverses sources, telles que les publications en accès libre ou les archives numériques. Ces données alimentent des études quantitatives ou qualitatives dans de multiples disciplines **[31].**

B) Limites et Considérations Ethiques

La collecte de données web soulève des enjeux importants en matière de conformité légale et d'éthique. Bien que l'accès à l'information soit essentiel pour la transparence, des réglementations comme la **loi n°2008-12 du 25 janvier 2008 portant sur la protection des données à caractère personnel au Sénégal** et le **RGPD** en Europe imposent des restrictions rigoureuses sur la collecte et l'utilisation des données personnelles **[32].**

Les conditions d'utilisation des sites web**,** les pratiques de collecte doivent être conformes aux termes et politiques des sites concernés pour éviter tout litige, blocage ou sanction.

La confidentialité et anonymisation des données**,** la collecte de données doit être réalisée en respectant strictement les réglementations sur la vie privée. Cela inclut l'obligation d'anonymiser les données pour garantir qu'aucune information collectée ne compromet les droits ou la confidentialité des individus **[30].**

1.1.2. Web Scraping

1.1.2.1. Définition

Le web scraping, ou extraction de données web, est une méthode automatisée permettant de collecter des informations disponibles sur les pages web. Cette technique s'appuie sur des programmes informatiques spécialisés, appelés scrapers, qui parcourent systématiquement les sites pour identifier des contenus spécifiques comme du texte, les images, les données tabulaires, etc. et les convertir en formats structurés tels que des fichiers CSV**,** JSON, ou des bases de données relationnelles. Contrairement à la collecte manuelle, le web scraping offre une solution rapide et scalable pour gérer de vastes volumes de données.

1.1.2.2. Fonctionnement

Le web scraping est une méthode automatisée permettant d'extraire des données depuis des sites web. Il utilise des scripts ou des logiciels pour simuler la navigation web humaine, accéder aux pages, analyser leur contenu et collecter les informations ciblées. Bien que sa mise en œuvre puisse être complexe, le processus peut être décomposé en plusieurs étapes clés :

- L'identification des sources c'est-à-dire la sélection des sites ou pages contenant les données souhaitées.

- La navigation automatisée qui équivaut à l'utilisation des scripts ou des outils pour parcourir les pages.
- L'analyse du contenu, c'est extraire les éléments pertinents du texte, images, données tabulaires.
- La structuration des données, consiste à l'organisation des informations collectées en formats exploitables (CSV, JSON, bases de données).

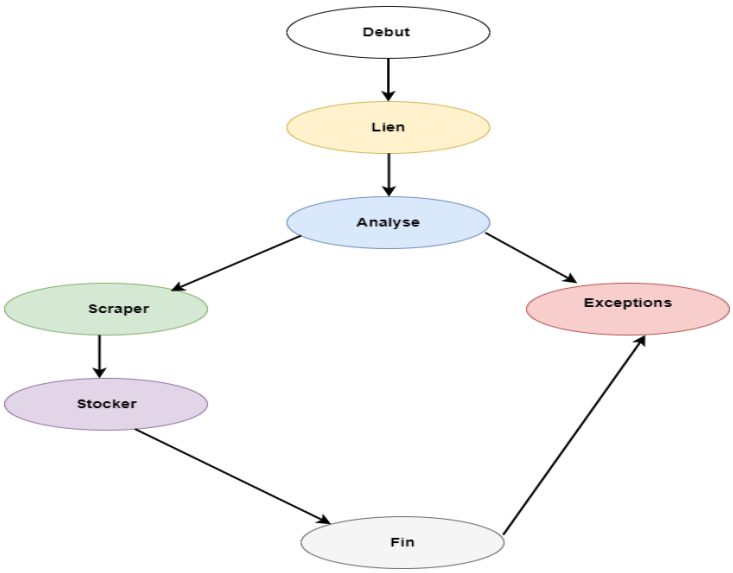


Figure 4: Algorithme de Scraping

A) Analyses et ciblages des données

La première étape du web scraping consiste à analyser la structure des pages web contenant les données recherchées. La majorité des sites utilisent des langages tels que HTML et CSS pour organiser et styliser leur contenu. Les développeurs doivent examiner le code source des pages à l'aide d'outils comme l'inspecteur d'éléments des navigateurs web. Ces outils permettent de localiser les informations pertinentes, souvent situées dans des balises spécifiques telles que ** , <table>,** ou **** **[33].**

B) Requête HTTP

Le web scraper envoie une requête http, du GET ou du POST au serveur hébergeant le site web pour accéder aux pages cibles. Cette étape reproduit le comportement d'un utilisateur humain entrant une URL dans son navigateur. Les scrapers utilisent souvent des bibliothèques comme Requests en Python pour automatiser cette interaction. Ces requêtes permettent de récupérer le contenu brut d'une page, généralement sous forme de HTML ou JSON **[34].**

C) Extraction et parsing des données

Après avoir récupéré la page, son contenu brut est analysé à l'aide de bibliothèques spécialisées. Ce processus, connu sous le nom de parsing, convertit le code HTML en un arbre DOM manipulable. Les scrapers peuvent alors naviguer dans cet arbre pour extraire les éléments d'intérêt, tels que des textes, des liens ou des images **[33].**

D) Structuration des données

Les données extraites sont ensuite nettoyées et organisées dans des formats structurés tels que CSV**,** JSON ou insérées directement dans une base de données. Cette étape est essentielle pour garantir que les informations soient facilement exploitables pour des analyses ultérieures ou des intégrations dans des systèmes tiers [34].

E) Gestion des restrictions et limitations

Les sites web mettent souvent en place des mécanismes pour limiter ou empêcher le scraping, tels que les CAPTCHA, les vérifications d'adresse IP, ou des limites de fréquence des requêtes. Pour surmonter ces obstacles, les scrapers peuvent adopter plusieurs stratégies, notamment l'utilisation des proxys ou des réseaux VPN pour changer régulièrement leur adresse IP et éviter d'être bloqués., l'intégration des solutions anti-CAPTCHA pour automatiser la résolution des tests, la régulation de la fréquence des requêtes afin de respecter les limites imposées par les serveurs et éviter de surcharger les systèmes.

Cependant, ces pratiques doivent être appliquées avec précaution pour éviter d'enfreindre les lois en vigueur ou les conditions d'utilisation des sites web [34].

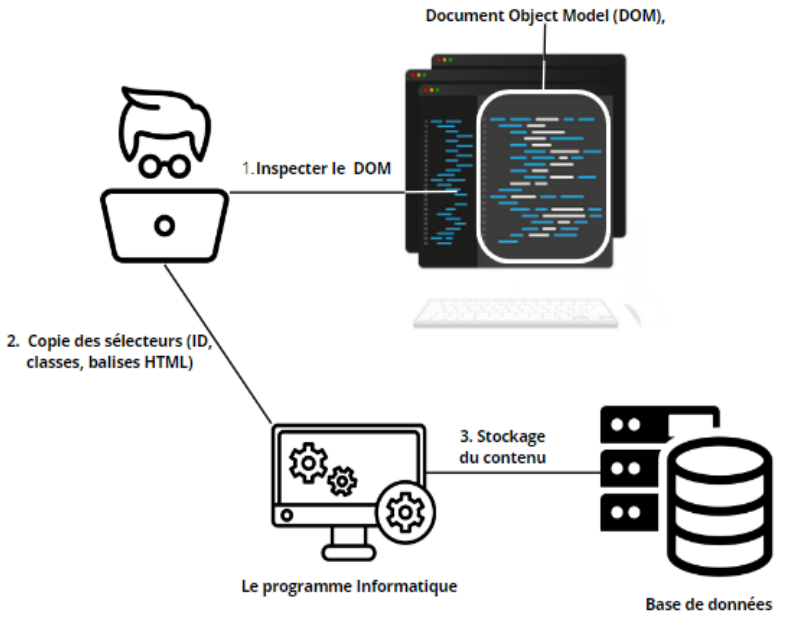


Figure 5: Processus de scraping complet

1.1.2.3. Formes de web Scraping

Le web scraping se décline en plusieurs types, selon les approches employées pour collecter les données et les objectifs spécifiques des utilisateurs. Ces méthodes varient par leur complexité**,** leur portée, et leur applicabilité à différents cas d'usage.

A) Scraping Statique

Le scraping statique se concentre sur l'extraction de données à partir de pages web fixes et non interactives. Ces pages, entièrement chargées dès qu'elles sont demandées, offrent un contenu directement accessible à l'aide d'outils simples tels que des requêtes HTTP et des bibliothèques de parsing HTML. Cette méthode est particulièrement adaptée pour des applications telles que la collecte d'informations sur des sites d'actualités ou des blogs, ou encore l'extraction de catalogues produits à partir de bases de données statiques. Parmi ses avantages, on note sa simplicité de mise en œuvre et son faible besoin en ressources, en comparaison avec des techniques plus complexes. Toutefois, le scraping statique présente aussi des limites, notamment son incapacité à traiter les sites dynamiques ou interactifs générés par JavaScript ou des Framework modernes **[35].**

B) Scraping Dynamique

Le scraping dynamique est une technique utilisée pour extraire des données provenant de pages web dont le contenu est généré ou modifié dynamiquement en temps réel par des scripts JavaScript ou des requêtes AJAX. Contrairement au scraping statique, cette méthode nécessite des outils capables de simuler un navigateur web, pour interagir avec le site et charger les données générées. Elle est particulièrement utile pour des applications telles que l'extraction de résultats en temps réel à partir de moteurs de recherche ou l'analyse des prix sur des plateformes d'e-commerce comme Expat-Dakar. Le principal avantage du scraping dynamique est qu'il permet d'accéder à des données qui ne sont pas directement visibles dans le code source initial de la page et qu'il gère efficacement les interactions complexes des sites web modernes. Cependant, cette méthode consomme davantage de ressources, car elle implique de charger et de rendre les pages comme le ferait un navigateur réel, et elle est également plus sujette à des mécanismes de détection et de blocage par les sites ciblés, tels que les CAPTCHA ou les restrictions d'adresse IP **[36].**

C) Scraping Focalisée

Le scraping ciblé (focalisé) vise à extraire des données spécifiques à partir de sites ou de pages web prédéfinis. L'objectif principal est d'identifier et de collecter des éléments précis, tels que des prix, des images, ou des coordonnées. Cette méthode est particulièrement utile pour des applications telles que le suivi des prix de produits sur des plateformes concurrentes ou la

collecte d'avis et de commentaires sur les réseaux sociaux. Parmi ses avantages, le scraping ciblé se distingue par sa rapidité et son efficacité, car il se concentre uniquement sur les informations nécessaires, ce qui réduit considérablement le volume des données collectées. Cependant, il présente aussi des limites, notamment la nécessité d'une configuration initiale détaillée pour cibler les bons éléments et une faible flexibilité face aux modifications structurelles des sites web **[37].**

D) Scraping Etendu

Le scraping étendu est une approche ambitieuse et puissante qui cible la collecte de vastes quantités de données provenant de nombreuses pages ou sites web. Contrairement à d'autres méthodes plus spécifiques, il excelle dans les projets nécessitant un volume massif de données, comme des analyses de marché globales ou des recherches académiques à grande échelle. Ses applications incluent la création de bases de données consolidées couvrant un secteur entier ou l'analyse comparative approfondie de milliers de produits ou services.

Ce type de scraping a des avantages indéniables, il permet de produire des ensembles de données riches et variés, indispensables pour une compréhension complète et détaillée des tendances ou des comportements. Cependant, il n'est pas sans limites. En collectant de manière intensive, cette méthode peut facilement dépasser les restrictions imposées par les sites web, exposant ainsi le scraper à des mécanismes de blocage. Pire encore, elle peut violer les conditions d'utilisation des plateformes, soulevant des questions légales et éthiques cruciales **[38].** Le scraping étendu n'est pas une stratégie pour les projets mineurs ou mal préparés : il exige une expertise technique et une vigilance rigoureuse pour rester conforme aux normes en vigueur.

1.1.2.4. Domaines d'utilisation

Le web scraping permet de récupérer des données provenant de divers sites ou plateformes web pour les stocker dans une base de données ou une feuille de calcul. Ces données, une fois extraites, peuvent être rapidement analysées et utilisées dans divers contextes. Le processus repose sur deux outils principaux : un crawler et un scraper. Le crawler agit comme un robot chargé de télécharger systématiquement les données d'internet, tandis que le scraper extrait les informations importantes sous une forme brute et les organise dans un format ou une structure prédéfinis, comme un fichier CSV ou une base de données.

Cette méthode est particulièrement utile dans des cas tels que la surveillance des prix des denrées alimentaires**,** où elle permet d'analyser les offres et les prix des produits disponibles sur le marché. De plus, le web scraping peut être appliqué au darknet**,** offrant des insights sur des activités illicites, comme les ventes de produits interdits ou les trafics de drogues.

Le web scraping est largement utilisé dans des domaines technologiques tels que la Business Intelligence**, l'**Intelligence Artificielle (IA**)**, la Data Science, le Big Data, le Cloud Computing, et la Cyber Security**.** Parmi ses applications les plus courantes figurent le suivi des prix, les études de marché, et l'analyse des sentiments. Par exemple, dans le secteur du commerce électronique, où le prix joue un rôle crucial, les entreprises surveillent étroitement les tarifs de leurs concurrents. Le web scraping automatise cette tâche, offrant des données précieuses pour ajuster les stratégies tarifaires et analyser les tendances du marché.

En outre, le web scraping peut être utilisé pour visualiser les changements boursiers, suivre l'évolution des prix au fil du temps, et analyser les commentaires ou les flux des médias sociaux. Ces informations permettent aux analystes et aux professionnels de l'informatique décisionnelle de prendre des décisions éclairées et de mieux comprendre les dynamiques du marché.

1.1.3. Web Crawler

1.1.3.1. Définition

Dans l'ère actuelle de l'économie de la donnée, les entreprises doivent exploiter des quantités massives de données, dont l'analyse peut significativement améliorer leur efficacité et leur rentabilité [8]. Au cours des dernières années, la technologie financière (Fin Tech) a connu une croissance notable grâce à l'impact du Big Data, tout en suscitant des débats sur la conformité légale et éthique [8]. Mais pour pouvoir utiliser ces données massives, il est essentiel de comprendre sa provenance et la manière dont elles ont été collectées.

Le Web crawler, également appelé Web spider ou Web robot, est un outil indispensable dans ce processus. Il s'agit d'un programme ou script automatisé conçu pour capturer les données publiées sur Internet selon des règles prédéfinies. Sa fonction principale est de parcourir automatiquement le réseau, de recueillir des informations et de construire un index, facilitant ainsi l'accès à ces données pour diverses applications.

Les Web crawlers jouent un rôle crucial dans l'infrastructure numérique moderne. Par exemple, des outils comme Google bot de Google, et Apple bot de Apple sont des exemples emblématiques de crawlers largement utilisés pour alimenter les moteurs de recherche. En effet, sans les Web crawlers, les moteurs de recherche ne pourraient pas fonctionner, car ils dépendent de ces outils pour indexer les milliards de pages accessibles sur Internet.

1.1.3.2. Fonctionnement d'un crawler web

Un web crawler est un programme automatisé dont les tâches sont prédéfinies pour explorer et indexer des pages web. Les comportements des crawlers peuvent varier selon les moteurs de recherche, mais ils suivent généralement un processus standardisé qui sont :

- Le processus commence par le téléchargement du fichier robots.txt du site web cible. Il indique aux robots quelles parties du site sont accessibles et quelles sections doivent être évitées, selon les directives des administrateurs web.
- **U**ne fois le fichier robots.txt consulté, le crawler explore les premières pages d'un site en fonction des URL prédéfinies ou accessibles via le fichier. Chaque page analysée fournit des informations sur son contenu et ses liens internes.
- Pendant l'exploration, le robot identifie de nouveaux liens présents sur les pages. Ces URL découvertes sont ajoutées à une file d'attente pour une exploration ultérieure.
- Le crawler continue à explorer les pages de la file d'attente, découvrant et ajoutant de nouvelles URL. Ce processus se poursuit jusqu'à ce que toutes les pages indexables soient explorées ou que le robot atteigne une page sans liens supplémentaires.

Ce fonctionnement itératif permet aux crawlers de construire une carte complète des pages et des liens d'un site web, facilitant l'indexation et l'organisation des contenus pour les moteurs de recherche.

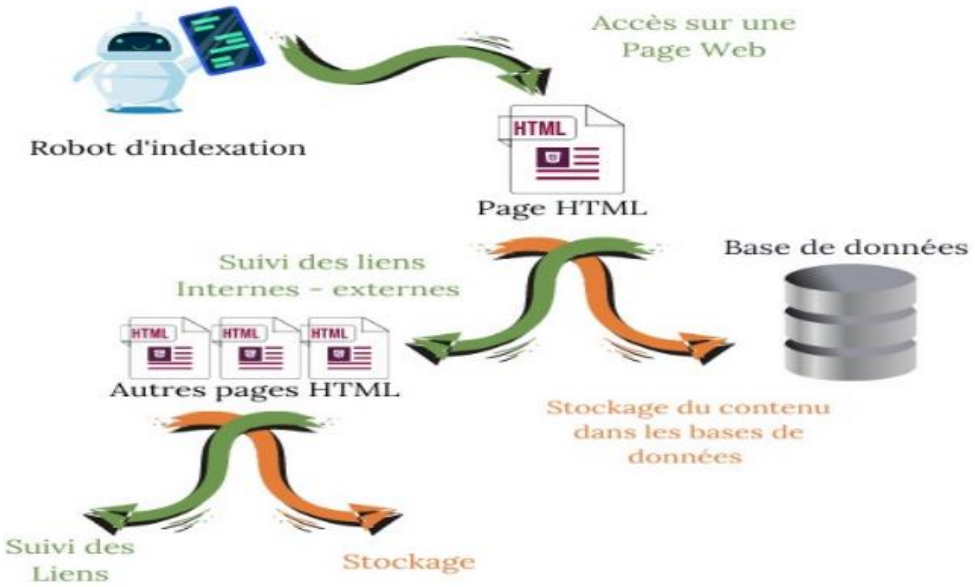


Figure 6: Processus de fonctionnement d'un crawler

1.1.3.3. Formes de web Crawling

A) Le web Crawling manuel

Le crawling manuel consiste à détecter et collecter des données ou des liens web directement par l'utilisateur, sans l'assistance de programmes automatisés. Cette méthode est généralement utilisée lorsque l'utilisateur a un besoin minimal ou un objectif précis et unique, tel que la récupération d'un ou deux liens spécifiques. Dans ces cas, le processus ne prend que quelques minutes et peut être efficace pour des tâches ponctuelles et limitées **[25].**

Cependant, cette approche présente des limites importantes lorsqu'il s'agit de collecter un grand nombre de liens ou de données suivant un prédicat bien défini. En effet, le crawling manuel devient rapidement fastidieux, chronophage et sujet à des erreurs humaines. Pour des tâches impliquant des volumes importants de données ou des règles complexes, cette méthode est inefficace et peu fiable, rendant nécessaire l'utilisation de crawlers automatisés. Ceux-ci permettent de gérer efficacement les besoins de collecte massive, tout en minimisant les erreurs et en optimisant le temps consacré à l'extraction.

B) Le Web Crawling a but general (Full crawling)

Le full crawling, contrairement au crawling manuel, repose sur un robot automatisé capable de collecter les liens et le contenu des pages web de manière exhaustive. Ce processus consiste à explorer toutes les URL spécifiées ainsi que leurs hyperliens, sans se limiter à des critères ou des filtres spécifiques. Bien que cette méthode permette de construire une base de données complète et large, elle est très coûteuse en termes de bande passante, de temps et de ressources réseau, car elle n'implique aucun ciblage sélectif **[23] [28].** Un exemple de full crawling est illustré par le fonctionnement de Google bot, le robot d'exploration de Google. Par exemple, lorsqu'un utilisateur recherche un article sur l'ancien Président Macky Sall, Google bot parcourt un très grand nombre de pages web liées à cette recherche. En suivant les hyperliens associés, il indexe les pages pour créer une vaste base de données accessible via Google. Le résultat est une liste complète et diversifiée de contenus provenant de plusieurs sites d'actualités sénégalais. Cependant, cette méthode consomme beaucoup de bande passante et de ressources, car elle n'applique pas de restrictions spécifiques sur les types de pages collectées. Pour réduire ces coûts et améliorer l'efficacité, une alternative consiste à utiliser un crawler ciblé, qui se concentre uniquement sur des pages répondant à des critères précis.

C) Le Web Crawling ciblé

Un crawler ciblé est conçu pour collecter des documents sur un sujet spécifique, réduisant ainsi le trafic réseau et les téléchargements inutiles. Contrairement au full crawling, ce type de robot

d'exploration se concentre uniquement sur les pages correspondant à un ensemble prédéfini de sujets. En limitant son exploration aux régions pertinentes du web, il permet des économies significatives en termes de matériel et de ressources réseau. L'objectif principal d'un crawler ciblé est d'offrir une collecte optimisée et sélective des données. Cela repose sur des mots-clés, des documents associés à une taxonomie de sujets, ou des ontologies pour définir les zones d'intérêt. Ces robots garantissent un bon équilibre entre les coûts, la couverture, et la qualité de la collecte en explorant efficacement les ressources pertinentes. Un des défis majeurs pour les crawlers ciblés est la hiérarchisation des URLs à visiter, qui peut être basée sur la distance des hyperliens : priorisation des pages proches en termes de liens ou les combinaisons d'analyse de texte et d'hyperliens : utilisation de techniques avancées, comme l'indexation sémantique latente, pour évaluer la pertinence des pages.

Plutôt que de s'appuyer sur la structure des applications web, les crawlers ciblés influencent leur comportement en fonction du contenu des pages web. Cela rend leur exploration plus pertinente et ciblée, en identifiant uniquement les informations spécifiques à leurs objectifs.

D) Le crawling distribué

Le crawling distribué repose sur l'utilisation de plusieurs processus répartis sur des machines ou des nœuds de calcul distincts pour explorer et télécharger des pages web. Cette approche permet d'améliorer l'efficacité et la rapidité de l'exploration en divisant les tâches entre plusieurs systèmes, plutôt que de s'appuyer sur un seul système centralisé. Elle est particulièrement adaptée pour gérer la vaste échelle du web moderne et répondre aux contraintes en termes de temps et de ressources.

Dans le cadre du crawling distribué, chaque nœud du réseau joue un rôle spécifique, et les tâches sont réparties pour optimiser l'exploration. Le partage de charge des nœuds partage la responsabilité de l'exploration de différentes parties du web, permettant une couverture plus rapide. Aussi l'indexation distribuée, chaque nœud contribue à la création d'un index global, mais ne conserve qu'une partie des données, minimisant ainsi les besoins en stockage pour chaque machine.

1.1.3.4. Domaines d'utilisation

Le web crawling est une technologie clé utilisée dans de nombreux domaines, grâce à sa capacité à explorer, collecter et indexer de grandes quantités de données web.

Dans l'e-commerce, le web crawling facilite la comparaison des prix et des promotions en temps réel. Les plateformes comme Jumia sont souvent scrutées pour détecter les variations de prix et surveiller les commentaires des utilisateurs [41]. Cela aide les entreprises à optimiser leurs offres et à répondre rapidement aux besoins des clients.

Les chercheurs l'utilisent aussi pour collecter des données scientifiques ou des publications académiques accessibles en ligne. Des plateformes comme Google Scholar[4] peuvent être explorées pour récupérer des articles en libre accès, contribuant à des études quantitatives ou qualitatives **[42].** Cela facilite la création de bases de données pertinentes pour des projets de recherche.
Dans les domaines des médias et de l'actualité, le crawling permet de collecter des données issues de sites d'information ou de réseaux sociaux. Les entreprises utilisent ces données pour suivre les sujets tendances et analyser les sentiments du public à propos d'événements récents **[43].** Cela contribue également à la veille médiatique et à la gestion de la réputation en ligne.

Le web crawling joue un rôle clé dans la cybersécurité, notamment pour la surveillance du darknet et la détection d'activités illicites, comme les ventes de données volées ou de produits illégaux **[44].** Les crawlers peuvent aussi identifier des vulnérabilités sur les sites web, aidant à prévenir des cyberattaques. **[45].** Dans le cadre de la data science, le web crawling est utilisé pour recueillir des avis clients ou des publications sur les réseaux sociaux. Ces données sont ensuite analysées pour déterminer les opinions ou les émotions associées à une marque, un produit ou un événement **[46].** Cette analyse aide les entreprises à adapter leurs stratégies marketing. Les entreprises exploitent le web crawling pour extraire des informations stratégiques. Ces données, souvent issues de concurrents ou de tendances du marché, permettent de guider les décisions commerciales et de détecter de nouvelles opportunités **[47].** Cela contribue à améliorer leur compétitivité sur le marché. Les données collectées grâce au web crawling permettent de mieux comprendre les préférences et les comportements des utilisateurs. Ces informations sont ensuite utilisées pour personnaliser les campagnes publicitaires, augmentant ainsi leur efficacité **[48].** Cela offre aux entreprises un avantage concurrentiel dans la création d'une expérience utilisateur sur mesure.

1.2. Etude comparative: Web Crawling Vs Web Scraping

Le **Web scraping** est une méthode d'extraction de données spécifiques à partir de sites web, qui sont ensuite stockées dans un format structuré comme des tableaux, des fichiers **JSON**, des bases de données, ou des fichiers XML. Il est principalement utilisé pour des applications ciblées, comme la comparaison de prix ou la collecte d'informations pour des analyses spécifiques. Les outils de scraping sont capables de contourner certaines restrictions, comme

[4] Un moteur de recherche qui indexe principalement le contenu de périodiques et de livres électroniques de nature universitaire provenant de plusieurs éditeurs commerciaux, sociétés savantes et universités.

celles définies dans les fichiers robots.txt, de se faire passer pour des navigateurs web, de soumettre des formulaires, et même d'exécuter du JavaScript pour simuler un comportement humain.

En revanche, le Web crawling est une méthode d'exploration systématique et automatisée du web. Il commence par une liste prédéterminée d'URL et cherche à suivre les liens pour explorer et indexer le contenu des sites web. C'est ce que fait un moteur de recherche comme Google, lorsqu'il visite et catalogue des pages web pour répondre aux requêtes des utilisateurs. Le principal objectif du crawling est de comprendre le web en explorant son contenu de manière exhaustive.

	Web crawling	**Web scraping**
But	Indexer et comprendre e contenu du web pour permettre aux utilisateurs d'extraire des informations.	Extraire des données spéci- ques à partir de sites web ciblés et les transformer en formats structurés comme des tables ou des bases de données.
Fonctionnement	Un crawler explore le web de manière systématique à partir d'une liste de URLs semences, en suivant les liens pour découvrir et indexer des pages.	Les scraping analysent le contenu des pages web pour extraire des données spéci ques définies à l'avance.
Exemple	Utilisé par les moteurs de recherche comme Google pour indexer le contenu du web et parfois par des outils de surveillance poursuivre les changements sur les sites.	Utilisé pour collecter des informations telles que les prix pour la comparaison, les avis de produits, etc.
Différences	Le crawling vise à explorer et indexer tout le contenu disponible. Pour le crawling, on part souvent d'URLs semences pour découvrir de nouvelles pages. Les crawlers respectent généralement robots.txt et se présentent comme des agents de recherche.	Tandis que les scraping se concentrent sur des données spécifiques sur des sites précis. Pour les scraping, on connaît généralement les sites cibles et les pages. Les scrapers peuvent ignorer ces directives pour accéder à des données spéci ques, parfois en simulant le comportement humain.

Tableau 1: Tableau comparatif du scraping et du crawling

1.3. Problématique

Dans un monde où la quantité d'informations disponibles en ligne croît de manière exponentielle, la capacité à collecter, analyser et exploiter ces données devient encore plus critique. Bien que certains outils actuels soient efficaces dans des domaines spécifiques, ils présentent d'importantes lacunes dans d'autres aspects. Ainsi, le véritable défi réside dans

l'adaptabilité des outils existants aux besoins spécifiques des utilisateurs, tout en respectant les exigences techniques, éthiques et légales.

1.3.1. Etude des besoins

La conception de crawlers et scrapers intelligents repose sur une compréhension approfondie des besoins des utilisateurs finaux. Dans cette optique, les spécifications liées à la collecte de données doivent être plus détaillées et sophistiquées. Les utilisateurs attendent alors des solutions adaptées à une large gamme de données et à l'évolution constante des environnements dans lesquels elles s'insèrent.

- ✓ **Besoin 1 : Identification des informations pertinentes**

Un besoin primordial consiste à définir avec précision les types de données à extraire. Cela englobe, les données textuelles, c'est-à-dire les articles de blogs, les descriptions de produits, les textes légaux, etc. Ces informations sont indispensables dans des domaines variés, allant de l'e-commerce à l'analyse juridique. Les images dans ces cas sont particulièrement utiles dans des secteurs comme le commerce électronique ou l'analyse multimédia. Les données sont structurées dans des tableaux, graphiques, ou formats JSON/XML, essentiels pour organiser et interpréter efficacement des volumes de données complexes.
La capacité d'un système à distinguer ces catégories et à les formater de manière exploitable est cruciale pour optimiser les processus d'analyse et de prise de décision.

- ✓ **Besoin 2 : Adaptation aux domaines d'application**

Chaque domaine impose des exigences spécifiques si on considère le domaine des E-commerce, les entreprises souhaitent surveiller les prix, les avis clients et les tendances en temps réel pour ajuster leurs stratégies commerciales et rester compétitives. Dans la recherche académique, elles, les chercheurs doivent accéder à des publications disponibles sur des plateformes ouvertes pour enrichir leurs travaux et étendre la portée de leurs recherches. Quant aux médias et réseaux sociaux, l'analyse des opinions publiques et des sujets d'actualité est essentielle pour comprendre les dynamiques sociales ou pour surveiller la réputation en ligne. Cette diversité d'applications souligne la nécessité de solutions flexibles et sur-mesure, adaptées à des contextes et des besoins variés. Cette diversité d'applications souligne la nécessité de solutions flexibles et sur-mesure, adaptées à des contextes et des besoins variés.

- ✓ **Besoin 3 : Contraintes de performance**

Les utilisateurs exigent des outils performants répondant aux critères comme la rapidité, la capacité à collecter et traiter de grands volumes de données en un temps réduit, indispensable dans des environnements où la réactivité est clé. Avec une précision, les données collectées doivent être pertinentes et alignées avec les objectifs définis par l'utilisateur. Et aussi l'adaptabilité, qu'il soit possible de fonctionner efficacement avec des sites web dynamiques ou aux structures évolutives, souvent générées par des technologies comme JavaScript.

- ✓ **Besoin 4 : Considérations éthiques et légales**

Dans un environnement où la réglementation des données personnelles se renforce, il est impératif de respecter les cadres juridiques et éthiques. Cela implique la conformité aux règles des sites web, respecter des fichiers robots.txt, définissant les zones accessibles ou restreintes pour les crawlers, aussi le respect du RGPD en Europe pour garantir que les données personnelles collectées soient anonymisées et utilisées de manière éthique et responsable.
L'équité et transparence, pour bannir les pratiques contraires à l'éthique, telles que le scraping abusif ou non autorisé, qui pourrait compromettre la confiance et la crédibilité des systèmes.
Ces besoins mettent en lumière l'importance de développer des solutions technologiques à la fois performantes et responsables, capables de répondre aux défis modernes de manière durable.

1.3.2. Etude de l'existant

Dans un univers numérique en perpétuelle évolution, les outils actuels de collecte et d'analyse de données, bien qu'ayant connu des avancées notables, demeurent limités sur plusieurs aspects. Ces lacunes entravent leur adoption à grande échelle, notamment dans des applications où une précision accrue, une adaptabilité dynamique et un respect rigoureux des réglementations sont indispensables. Les web crawlers traditionnels, bien que performants pour explorer et indexer une grande quantité de pages, montrent leurs faiblesses dans plusieurs contextes que ça soit dans la consommation excessive de ressources avec des outils avec le parcoure systématique de toutes les pages sans discrimination, ce qui consomme une grande quantité de bande passante et de puissance de calcul, souvent inutilement.
Aussi avec les résultats qui sont peu pertinents avec l'absence de ciblage précis entraîne une collecte massive de données non exploitables ou hors contexte, compliquant leur utilisation.
L'incapacité à gérer les données dynamiques, de nombreux sites utilisent des technologies comme JavaScript ou AJAX pour générer du contenu en temps réel, ce qui échappe aux scrapers manuels sans outils avancés. Les barrières techniques, avec certains sites mettent en place des

obstacles tels que les CAPTCHA ou des limites sur les requêtes répétées, ralentissant ou bloquant ces scrapers basiques. Le manque de personnalisation, des outils sont conçus pour une couverture générale et ne permettent pas de stratégies d'exploration adaptées à des besoins spécifiques, comme l'analyse sectorielle ou la collecte ciblée. Il y a aussi les modifications fréquentes des structures des sites qui mettent régulièrement à jour leurs interfaces et architectures pour améliorer l'expérience utilisateur, rendant l'extraction de données automatisée plus complexe.

Chapitre 2 : Etat de l'art et positionnement

2.1. Etat de l'art

2.1.1. Scraping intelligent

Le scraping intelligent est une méthode avancée et automatisée d'extraction de données web, intégrant des technologies comme l'intelligence artificielle et le traitement du langage naturel. Ces outils permettent de personnaliser les processus d'extraction en fonction des structures dynamiques et complexes des sites web, tout en réduisant au minimum l'intervention humaine. Dans le contexte du Big Data, ces technologies se sont imposées comme indispensables pour collecter et analyser de vastes volumes de données de manière rapide et précise. Les domaines d'application incluent l'analyse de marché, la personnalisation des services numériques et la recherche scientifique. De plus, le scraping intelligent permet de surmonter certaines limites des approches traditionnelles en automatisant l'adaptation aux changements structurels des pages web.

Cependant, son utilisation soulève diverses questions éthiques et juridiques, notamment en ce qui concerne la protection des données personnelles et le respect des droits d'accès. Cela implique une réflexion approfondie pour garantir une utilisation responsable, conforme aux cadres réglementaires.

2.1.1.1. Techniques modernes de scraping intelligent

A. Scraping basé sur l'apprentissage automatique (Machine Learning)

L'intégration de l'apprentissage automatique dans le domaine du scraping web permet de franchir les limites des approches traditionnelles en automatisant le traitement et l'extraction des données sur des pages web complexes. Les réseaux neuronaux, tels que les réseaux de neurones artificiels (ANN) et les réseaux à mémoire à long terme (LSTM), se distinguent par leur capacité à analyser et à s'adapter aux structures dynamiques des sites web. À titre d'illustration, les modèles de Deep Learning associés au traitement automatique du langage naturel offrent des outils puissants pour interpréter les contextes sémantiques et extraire des informations pertinentes, même dans des environnements peu structurés. Une étude récente a mis en évidence que ces modèles, en combinant les atouts du NLP et du Deep Learning, optimisent les processus d'extraction de données, les rendant ainsi plus performants et flexibles **[49].**

B. Scraping orienté modèle

Le scraping orienté modèle repose sur l'utilisation de templates ou de modèles prédéfinis pour naviguer sur des sites web spécifiques et en extraire les données souhaitées. Contrairement aux méthodes plus générales ou basées sur des règles, cette approche est particulièrement adaptée aux environnements où les sites web ont des structures relativement stables mais susceptibles de subir des modifications ponctuelles. Ces modèles permettent non seulement de guider le processus d'extraction, mais aussi de détecter et de corriger automatiquement les changements structurels des pages web. Ainsi, ils réduisent considérablement le besoin de mises à jour manuelles, souvent fastidieuses et chronophages.

Une étude marquante a introduit un algorithme innovant capable de mettre à jour automatiquement les paramètres de scraping en réponse aux modifications dans la structure des sites web **[50].** Si un site modifie la disposition de ses menus, des balises HTML ou d'autres éléments critiques, cet algorithme ajuste en temps réel les modèles pour garantir que l'extraction des données ne soit pas interrompue. Cette capacité est particulièrement utile dans des secteurs où les données évoluent rapidement, comme le commerce électronique, les services financiers, ou encore les plateformes d'agrégation d'informations.

L’approche orientée modèle présente des avantages remarquables de par sa robustesse. Il a apporté une amélioration considérable sur la stabilité des processus d’extraction, même dans des environnements de web dynamiques. Il offre également une précision de par sa capacité à s’adapter automatiquement aux nouvelles structures de page web. Ceci réduit le risque d'extraire des données incorrectes ou incomplètes. Il y a aussi la réduction des couts qui étaient consacrés à la mise à jour manuelle des configurations. Cela permet une gaies en termes de temps et de ressources humaines.

Il est essentiel de rappeler que cette approche présente également des défis à surmonter. Ces algorithmes orientés modèles nécessitent souvent une phase initiale de configuration rigoureuse pour définir les modèles et les paramètres. Ils peuvent aussi rencontrer des difficultés face à des sites web extrêmement dynamiques ou très non structurés.

C. Scraping basé sur le traitement du langage naturel (NLP)

Le scraping basé sur du NLP ouvre de nouvelles perspectives dans l’extraction et l’analyse de données issues de contenus textuels non structurés présents sur le web. Contrairement aux méthodes traditionnelles d'extraction de données, qui se concentrent souvent sur des éléments purement syntaxiques ou structuraux, les techniques de NLP permettent de comprendre le contexte et la sémantique des informations. Cela est particulièrement utile dans des environnements complexes où le texte est riche en significations implicites ou sujet à des variations linguistiques.

Le NLP permet de traiter et d'analyser des données textuelles à un niveau plus profond. Il peut identifier des relations entre les mots, comprendre les phrases dans leur globalité, et détecter des entités nommées comme des noms de personnes, des lieux ou des organisations. En intégrant des modèles capables de traiter plusieurs langues, le NLP facilite l'extraction de données dans des environnements globaux, sur des sites utilisant différentes langues ou dialectes. Les algorithmes de NLP peuvent organiser et catégoriser automatiquement les données extraites, les rendant exploitables pour des cas d'utilisation spécifiques.

Dans une étude récente, il a été démontré comment le NLP peut être appliqué de manière innovante à des plateformes éducatives multilingues. Leur travail consistait à construire une ontologie multilingue **[51],** c'est-à-dire un cadre structuré qui relie les concepts pédagogiques à travers plusieurs langues ce qui a permis :

- De surmonter les barrières linguistiques, les étudiants et enseignants de différentes régions du monde ont pu accéder à des contenus pédagogiques dans leur langue native, rendant l'apprentissage plus inclusif.
- D'améliorer l'organisation des données pédagogiques, grâce à une compréhension contextuelle des termes et concepts, l'ontologie a facilité la collecte, le classement et l'analyse des informations éducatives provenant de différentes cultures et contextes linguistiques.

Cette méthode n'est pas sans défis. Les modèles de NLP doivent être entraines sur des ensembles de données massifs pour atteindre un haut niveau de précision, ce qui peut être couteux en termes de temps et de ressources. De plus, traiter des langues moins répandues comme le Wolof ou des variations culturelles peut être complexe.

2.1.1.2. Technologies et outils pour le scraping intelligent

A. Les Frameworks modernes

Les frameworks modernes tels que Beautiful Soup, Scrapy, et Sélénium jouent un rôle essentiel dans l'extraction automatisée de données web. Chacun de ces outils possède des caractéristiques uniques qui les rendent adaptés à différents types de tâches de scraping. Chacune d'entre elle a ses caractéristiques, ses avantages et ses limites.

Le Beautiful Soup**,** est une bibliothèque Python conçue pour extraire des données à partir de documents HTML et XML. Son principal atout est sa simplicité et sa flexibilité dans le traitement des structures web. Il permet de naviguer, rechercher et modifier facilement l'arborescence HTML/XML. Ce dernier fournit des outils simples pour trouver des balises, des classes, ou des identifiants spécifiques. Il peut disposes des outils pour effectuer des requêtes http comme requests. La bibliothéque Beautiful Soup est facile à apprendre et utiliser, idéal

pour les petits projets ou les débutants. Il fonctionne très bien avec des pages HTML statiques, et il a une faible consommation de ressources comparée à d'autres outils. Cependant il est moins adapté aux sites dynamiques qui utilisent JavaScript. Ce dernier ne gère pas le téléchargement ou la manipulation des fichiers automatiquement.

Le scrapy, est un Framework Python puissant et scalable pour le web scraping, conçu pour les projets de grande envergure. Il est souvent préféré pour ses performances et son approche orientée projet. Il offre une gestion intégrée des requêtes HTTP, des cookies et des sessions. Il dispose également d'un moteur d'exploration (crawler) efficace pour parcourir des pages en profondeur. Dans scrapy, nous avons des outils pour exporter les données dans des formats variés comme JSON, CSV, ou bases de données. L'avantage de scrapy est qu'il est très performant pour le scraping à grande échelle, il supporte le scraping des sites dynamiques et complexes. Il dispose d'un écosystème riche de plugins pour étendre des fonctionnalités. Malgré ces avantages, il présente des limites à savoir, il peut être difficile à adopter pour les débutants. A cela s'ajoute il requiert une configuration initiale plus complexe que Beautiful Soup.

Le Sélénium**,** est un outil d'automatisation qui simule un navigateur web. Bien qu'il soit principalement utilisé pour les tests automatisés, il est également adapté pour le scraping de sites web dynamiques. Sélénium permet de contrôler un navigateur réel (chrome, Firefox, …), rendant possible l'interaction avec des éléments JavaScript, des menus déroulants, et des formulaires. Il permet de gérer aussi les sites qui nécessitent une interaction avec l'utilisateur, comme le défilement ou le clic. L'avantage d'utiliser le Framework Sélénium est qu'il est parfait pour les sites utilisant du JavaScript ou qui nécessitent des interactions complexes. Il peut aussi manipuler des sites protégés par des captchas simples grâce à des extensions. En effet il est lent et gourmand en ressources que Beautiful Soup ou Scrapy. Il peut être plus compliqué à configurer pour des projets à grande échelle.

B. Principaux services cloud pour le scraping

L'intégration des services cloud dans les projets de web scraping représente une avancée significative pour gérer des tâches de collecte de données à grande échelle. Ces services permettent d'assurer la scalabilité, la fiabilité, et l'efficacité du processus d'extraction, tout en offrant des solutions économiques pour traiter d'énormes volumes de données en temps réel.

Le service d'Amazon Web Services, est une plateforme de calcul serverless qui exécute du code en réponse à des événements tout en gérant automatiquement les ressources nécessaires. Ce service offre une scalabilité automatique, permettant de traiter des milliers de requêtes simultanément sans intervention manuelle, ce qui le rend idéal pour les projets nécessitant une

grande flexibilité. De plus, son modèle de paiement à l'utilisation basé sur la durée d'exécution et les ressources consommées en fait une solution particulièrement économique. AWS Lambda s'intègre facilement avec d'autres services AWS tels que S3, pour le stockage des données, ou DynamoDB, pour la gestion des bases de données. Cela en fait un outil puissant pour des cas comme l'extraction de données en continu depuis plusieurs sites, ou encore le traitement et stockage immédiat des informations collectées dans des entrepôts de données.

La plateforme serverless Google Cloud Functions est comparable à AWS Lambda, et elle est conçue pour exécuter des scripts ou des fonctions, notamment dans le cadre de projets de scraping. Ce service se distingue par sa flexibilité linguistique, prenant en charge des langages variés tels que Python et Node.js, ce qui le rend accessible à de nombreux développeurs. Grâce à son intégration native avec les outils Google, il s'avère particulièrement adapter aux projets exploitant BigQuery pour l'analyse de données ou Cloud Storage pour leur stockage. De plus, il est parfaitement optimisé pour l'accès aux API, facilitant les interactions avec d'autres services en ligne. Google Cloud Functions est idéal pour l'automatisation de scraping sur des sites riches en contenu, ainsi que pour l'exécution programmée de tâches à l'aide de Google Cloud Scheduler, rendant le processus d'extraction de données fluide et efficace.

Quant à Microsoft Azure Functions qui est aussi une plateforme serverless mais qui offre une infrastructure puissante et flexible pour automatiser des tâches telles que le scraping web. Dotée d'une infrastructure robuste, elle est capable de gérer des charges de travail lourdes, garantissant une performance optimale pour des projets exigeants. Azure Functions s'intègre de manière fluide avec d'autres services de l'écosystème Azure, comme Azure Storage pour le stockage des données extraites et Cosmos DB pour leur gestion, simplifiant ainsi le traitement et l'organisation des informations. De plus, elle prend en charge des workflows complexes grâce à l'outil Azure Logic Apps, permettant une orchestration efficace des processus. Cette combinaison de puissance, de flexibilité, et d'intégration fait d'Azure Functions une solution idéale pour les projets d'extraction et d'analyse de données à grande échelle.

2.1.2. Crawling intelligent

Le crawling web intelligent se réfère à la capacité des programmes automatisés, appelés crawlers ou robots, à naviguer sur le web de manière optimisée et contextuelle pour collecter des données pertinentes à partir de vastes ensembles de pages web. Contrairement aux crawlers traditionnels, ces systèmes intelligents intègrent des techniques avancées, telles que l'intelligence artificielle (IA), l'apprentissage automatique, et des algorithmes heuristiques. Ces innovations permettent aux crawlers d'adapter dynamiquement leurs stratégies de navigation

en fonction des objectifs spécifiques, du domaine d'application, ou des contraintes imposées par les sites web.

2.1.2.1 Approches basées sur l'intelligence artificielle

A. Crawling intelligent basée sur l'apprentissage par renforcement

L'apprentissage par renforcement constitue une solution innovante pour optimiser les décisions prises par les crawlers web, en leur conférant une capacité d'adaptation et d'efficacité bien supérieure à celle des méthodes traditionnelles. Grâce à l'intégration de l'intelligence artificielle, ces systèmes sont entraînés à identifier et prioriser les pages web les plus pertinentes à explorer, en apprenant de leurs propres interactions avec le web.

Concrètement, les algorithmes de renforcement permettent aux crawlers de s'auto-ajuster en temps réel, en adaptant leurs stratégies à mesure qu'ils accumulent des données sur les résultats obtenus. Par exemple, en analysant les liens rencontrés sur les pages visitées, ces algorithmes sont capables de prédire lesquels mèneront à des informations utiles, réduisant ainsi les ressources gaspillées sur des contenus non pertinents. Cette capacité à optimiser la navigation rend ces crawlers particulièrement adaptés à des environnements complexes ou dynamiques.

Une application remarquable de cette approche est le crawling adaptatif basé sur les ontologies, où les préférences spécifiques d'un utilisateur ou les exigences d'un domaine particulier sont utilisées pour guider le processus d'exploration. Selon une étude **[52],** cette méthode permet aux crawlers de cibler efficacement les informations pertinentes en s'appuyant sur des structures sémantiques préalablement définies, démontrant ainsi comment l'apprentissage par renforcement peut transformer des outils de collecte de données en systèmes intelligents et adaptatifs, capables de répondre aux besoins évolutifs des utilisateurs.

Cette approche présente plusieurs limites importantes. Sa performance dépend largement de la qualité des ontologies, car des modèles incomplets ou mal structurés réduisent son efficacité. De plus, la création et la maintenance des modèles d'intérêt utilisateur impliquent une complexité élevée en termes de calcul et de gestion des données. L'approche est également limitée en évolutivité, étant mieux adaptée à des domaines spécifiques qu'à des applications nécessitant une exploration web exhaustive. Des considérations éthiques et légales émergent également, notamment concernant la confidentialité et la conformité aux réglementations. Et pour finir, cette méthode repose fortement sur des données historiques fiables : en cas de biais ou de données insuffisantes, la pertinence des résultats peut être compromise.

B. Prédiction de la pertinence des pages avec des réseaux neuronaux

La prédiction de la pertinence des pages web à l'aide de réseaux neuronaux est une approche puissante pour optimiser les processus de crawling intelligent. Les modèles de prédiction, tels

que les réseaux de neurones artificiels (ANNs), permettent d'évaluer la pertinence des pages en se basant sur des caractéristiques spécifiques, comme le contenu textuel, les métadonnées, ou les liens internes.

Un exemple notable est l'étude menée par Dhabliya et al. (2023), qui a développé un modèle basé sur un perceptron multicouche (MLP) pour classer les pages selon leur pertinence. Ce modèle a atteint une précision impressionnante de 93 %, démontrant l'efficacité des réseaux neuronaux pour améliorer la qualité et la vitesse des processus d'exploration. Cette approche peut être appliquée dans divers contextes, tels que l'analyse de contenu ou la veille stratégique, en réduisant le temps et les ressources consacrés à l'exploration de pages non pertinentes.

Les limites de cette approche incluent une forte dépendance à des données d'entraînement de haute qualité pour garantir la précision des prédictions, ce qui peut être difficile à obtenir pour des domaines variés. De plus, les modèles comme les réseaux de neurones artificiels nécessitent une puissance de calcul importante et peuvent être coûteux à entraîner et à déployer. Enfin, ils peuvent manquer de transparence dans leurs décisions, rendant difficile l'interprétation des résultats.

C. Résistance aux mesures anti-crawling

Pour contourner les CAPTCHAs et les blocages IP, plusieurs techniques avancées sont utilisées. L'intégration de proxys rotatifs permet de masquer l'adresse IP d'origine en alternant entre différentes adresses, réduisant ainsi le risque de détection. La gestion des user-agents consiste à simuler différents navigateurs ou appareils pour rendre les requêtes moins suspectes. Des outils comme Puppeteer sont également efficaces pour interagir avec les pages web de manière réaliste, imitant un comportement humain. Une étude a démontré que l'application combinée de ces méthodes améliore considérablement la résilience des crawlers face aux restrictions, permettant une collecte de données plus fiable et continue.

2.1.3. Approches mixtes

Les approches mixtes offrent une solution complète en réunissant la précision des crawlers intelligents pour naviguer sur les sites web et l'efficacité des techniques de scraping intelligent pour extraire les données. Ces systèmes hybrides permettent de cibler les informations pertinentes tout en assurant une extraction rapide et efficace, même face à des environnements web dynamiques ou particulièrement complexes.

2.1.3.1. Combinaison de crawling et de scraping intelligents

Les pipelines intégrés de collecte et d'extraction de données combinent harmonieusement les atouts du crawling intelligent et du scraping intelligent, offrant une solution complète et performante pour la gestion des données web. Le crawling intelligent explore les sites web et identifie les sections pertinentes, guidant ainsi le processus vers les pages contenant les informations recherchées. Une fois ces sections ciblées, le scraping intelligent prend le relais pour extraire les données spécifiques en s'appuyant sur des techniques avancées, comme le NLP ou des algorithmes prédictifs. Cette synergie permet de collecter des informations précises et détaillées, même dans des environnements complexes ou peu structurés. En outre, l'automatisation dynamique, rendue possible par l'apprentissage automatique, optimise ces approches mixtes. Ces systèmes intelligents s'adaptent en temps réel aux modifications des structures des sites web, garantissant une collecte continue, efficace et pertinente tout en réduisant les besoins d'interventions manuelles.

2.1.3.2. Technologies et outils pour les approches mixtes

La combinaison de Scrapy et Sélénium constitue une approche puissante pour le crawling et le scraping des données web en exploitant leurs forces complémentaires. Scrapy, réputé pour sa rapidité et son efficacité, est idéal pour l'exploration et la récupération de données à grande échelle grâce à son architecture asynchrone. Cependant, il atteint ses limites face aux sites dynamiques générés par JavaScript. Sélénium intervient alors pour simuler un navigateur réel et interagir avec ces pages complexes, offrant ainsi la capacité de traiter des sites riches en contenu dynamique. Ensemble, Scrapy et Sélénium permettent une exploration rapide et une extraction précise, combinant le meilleur des deux outils pour gérer efficacement les sites web modernes.

2.1.3.3. Limites et défis des approches mixtes

La conception et la maintenance des systèmes hybrides, qui combinent le crawling et le scraping, nécessitent une expertise technique avancée**.** L'intégration d'outils comme Scrapy, Sélénium, ou Apify, par exemple, exige une compréhension approfondie des architectures web et des environnements dynamiques. En outre, l'optimisation des performances, qui implique de maximiser la rapidité et la précision tout en réduisant les coûts, peut être un défi majeur. Cette complexité peut devenir encore plus grande lorsque des techniques avancées comme le machine Learning ou le NLP sont intégrées pour améliorer la collecte de données.

Les approches mixtes sont souvent gourmandes en ressources, en raison de leur double nature. Scrapy excelle dans le crawling rapide et efficace, mais l'ajout de Sélénium pour interpréter des

pages dynamiques augmente considérablement la consommation de temps de calcul et de mémoire. De plus, la bande passante nécessaire pour gérer un grand nombre de requêtes simultanées ou pour interagir avec des pages complexes peut rapidement devenir un goulot d'étranglement, en particulier pour les projets à grande échelle. Cela nécessite une infrastructure solide et, souvent, un investissement dans des solutions cloud pour garantir la scalabilité.

Les approches mixtes, tout comme les techniques de scraping et de crawling individuels, sont soumises à des règles strictes en matière de protection des données. Les systèmes hybrides doivent donc éviter les pratiques intrusives, telles que le scraping de données personnelles sans consentement, et respecter les conditions d'utilisation des sites web. En outre, ils doivent veiller à ne pas perturber le fonctionnement des serveurs cibles en envoyant trop de requêtes, ce qui pourrait être perçu comme une attaque. Ces considérations nécessitent une planification rigoureuse pour rester dans les limites légales et éthiques.

2.2. Positionnements

2.2.1. Rôle du crawling et scraping intelligents dans la collecte de données

Le crawling et le scraping intelligents jouent un rôle clé dans la collecte de données web en offrant une approche complémentaire et intégrée, capable de naviguer efficacement sur des sites complexes et d'extraire des informations pertinentes. Le crawling intelligent, basé sur des algorithmes avancés tels que l'apprentissage par renforcement (Reinforcement Learning), permet d'identifier les pages les plus pertinentes en optimisant les stratégies de navigation. En parallèle, le scraping intelligent, reposant sur des technologies sur NLP et les réseaux neuronaux profonds, extrait avec précision les données, même dans des environnements dynamiques ou non structurés. Cette synergie garantit une collecte de données ciblée, rapide et adaptable aux besoins spécifiques.

Dans cette approche hybride, les deux processus s'intègrent pour former un pipeline continu et automatisé. Le crawler intelligent explore les sites en identifiant les sections pertinentes grâce à des algorithmes comme le Deep Q-Learning, qui évalue dynamiquement la valeur des liens ou des pages en fonction des objectifs définis. Une fois ces pages détectées, le scraper intelligent intervient, utilisant des frameworks avancés tels que BERT pour extraire et organiser les informations tout en tenant compte du contexte sémantique. Ce modèle intégré s'appuie sur des systèmes d'apprentissage automatique qui ajustent automatiquement les stratégies de navigation et d'extraction en réponse aux changements structurels des sites web, comme la modification de balises HTML ou de menus.

Cette approche est particulièrement puissante grâce à ses choix d'algorithmes. Le Deep Q-Learning permet d'économiser les ressources en ciblant uniquement les contenus pertinents,

tandis que les modèles NLP, comme BERT, garantissent une extraction précise et contextuelle des données textuelles. Cette combinaison offre une solution robuste, résiliente et efficace, idéale pour des domaines tels que la veille stratégique, le commerce électronique et la surveillance des tendances, où les données évoluent rapidement et nécessitent des mises à jour fiables et fréquentes.

2.2.2. Positionnement vis-à-vis des Enjeux et Défis

Les approches traditionnelles de crawling et de scraping web présentent des limitations importantes qui réduisent leur efficacité dans des environnements modernes. Parmi ces défis, l'évolutivité est un problème majeur : les solutions classiques peinent à gérer de larges volumes de données, entraînant une surconsommation de ressources et des temps de traitement prolongés. De plus, les sites web dynamiques, souvent basés sur des frameworks JavaScript ou soumis à des modifications fréquentes de structure, rendent les systèmes traditionnels inefficaces pour l'extraction de données fiable. Enfin, des considérations éthiques et légales, notamment le respect des réglementations sur la protection des données, pose des contraintes supplémentaires.

Pour dépasser ces limitations, des solutions intégrant des technologies avancées ont été identifiées. L'évolutivité peut être améliorée grâce à l'utilisation de services cloud tels qu'AWS Lambda, qui offrent une montée en charge automatique et une gestion efficace des ressources. Ces architectures permettent de traiter des volumes de données massifs tout en optimisant les coûts et les performances.

Concernant les sites dynamiques, l'adoption de modèles d'intelligence artificielle, comme le Deep Q-Learning pour le crawling et les frameworks basés sur le traitement du langage naturel (NLP) comme BERT, offre une flexibilité accrue. Ces techniques permettent aux systèmes de s'ajuster automatiquement aux changements structurels des sites web, comme des modifications dans les balises HTML ou la réorganisation des éléments DOM, assurant ainsi une collecte continue et précise des données.

Sur le plan éthique et légal, des mécanismes peuvent être intégrés pour garantir la conformité avec des cadres réglementaires. Par exemple, des outils basés sur le NLP permettent d'identifier les données sensibles et de mettre en œuvre des techniques d'anonymisation ou de filtrage pour éviter le traitement non autorisé d'informations personnelles. Cela répond à la nécessité de collecter des données de manière responsable tout en respectant les droits des utilisateurs.

Chapitre 3 : Cahier des charges et choix technologies

3.1. Cahier des charges

3.1.1. Contexte du Projet

Avec l'augmentation exponentielle des données en ligne, la gestion et l'exploitation efficace de ces informations deviennent un défi de taille. Cela nécessite des outils capables non seulement de collecter et d'organiser ces données, mais aussi de les analyser de manière précise et en temps réel. En effet, les volumes gigantesques d'informations disponibles, souvent non structurées et en constante évolution, rendent les approches traditionnelles, comme les moteurs de recherche classiques, de moins en moins adaptées. Ces derniers, bien qu'efficaces pour des recherches basiques, peinent à répondre aux exigences croissantes d'analyse et d'interprétation des écosystèmes numériques modernes, qui sont à la fois complexes, dynamiques et souvent interdépendants.

C'est dans ce contexte que ce projet s'inscrit, avec pour ambition d'apporter une solution novatrice et performante. En développant un crawler et un scraper intelligents, dotés de technologies avancées reposant sur l'intelligence artificielle, ce projet vise à dépasser les limitations des outils existants. Ces technologies permettront non seulement de collecter des données de manière ciblée et efficace, mais aussi d'enrichir cette collecte grâce à des mécanismes d'analyse contextuelle et de reconnaissance des schémas complexes.

En intégrant l'intelligence artificielle, ce projet propose une véritable évolution dans le domaine de la gestion des données en ligne. Les outils conçus pourront s'adapter aux spécificités de chaque environnement numérique, identifier les informations pertinentes avec précision, et automatiser des tâches aujourd'hui complexes pour les utilisateurs. Ainsi, ce projet répond à une double exigence : accompagner la transformation numérique actuelle tout en permettant une utilisation optimisée et stratégique des données dans des environnements toujours plus compétitifs.

3.1.2. Objectifs du Projet

Le projet s'articule autour d'une série d'objectifs stratégiques et techniques, visant à répondre aux défis posés par l'explosion des données en ligne et à maximiser leur exploitation de manière innovante, performante et responsable.

L'objectif principal du projet est de développer un système capable d'automatiser la collecte d'informations issues du web de manière intelligente et adaptative. Cela inclut la création de solutions capables de naviguer de façon autonome dans des environnements numériques

complexes et dynamiques, en minimisant les interventions humaines et en garantissant une collecte rapide et précise.

Le projet vise également à intégrer des technologies avancées d'intelligence artificielle, le modèle de BERT et l'algorithme de Deep Q-Learning afin d'extraire des informations pertinentes et structurées à partir de grandes quantités de données brutes. Ces modèles permettront de détecter des schémas complexes, d'analyser des contenus contextuels, et de fournir des données prêtes à être exploitées dans divers cas d'usage.

Pour répondre à la volatilité et à la diversité des environnements web, le système sera conçu pour s'adapter automatiquement aux modifications structurelles des sites web. Cela permettra d'assurer une collecte précise, en évitant les ruptures ou les erreurs causées par les changements fréquents de formats ou d'architectures des pages en ligne.

Un point central du projet réside dans le respect des cadres éthiques et juridiques en vigueur. Cela inclut la conformité stricte au Règlement Général sur la Protection des Données ainsi qu'aux politiques spécifiques des sites web. Le système sera conçu pour minimiser les impacts intrusifs, respecter la confidentialité des données, et s'assurer que toutes les collectes soient effectuées dans le respect des droits des utilisateurs et des propriétaires de contenus.

Enfin, le projet prévoit la mise en place d'un prototype opérationnel, capable de s'adapter à des environnements numériques diversifiés et à des charges de travail croissantes. Ce prototype permettra une évaluation en conditions réelles, garantissant la validité des solutions proposées et leur capacité à répondre aux besoins du marché. La scalabilité sera au cœur de sa conception, pour permettre une montée en puissance progressive tout en maintenant une efficacité et une robustesse optimales.

3.1.3. Le périmètre

Le projet se focalise sur la collecte de données structurées et semi-structurées issues de pages web, en mettant l'accent sur la pertinence et l'efficacité des informations extraites. Il intègre également une analyse approfondie et une expérimentation de technologies avancées pour le crawling et le scraping intelligents, permettant d'optimiser les processus de navigation et d'extraction des données. L'objectif final est de valider un prototype fonctionnel évolutif, capable de s'adapter à des besoins variés, tels que la création d'une base de données historique dédiée au Sénégal**,** en collectant et en organisant des informations issues d'archives numériques, de publications et d'autres sources pertinentes. Ce projet démontre ainsi sa capacité à répondre à des cas d'utilisation concrets et à s'intégrer dans des environnements réels et diversifiés.

3.1.4. Etude de l'existant

Une analyse approfondie des outils actuels met en lumière plusieurs limitations majeures, une inadéquation face aux pages dynamiques utilisant des technologies modernes comme JavaScript, une gestion inefficace des restrictions imposées par les sites, telles que les CAPTCHA ou les directives des fichiers robots.txt, ainsi qu'une pertinence limitée des données collectées, souvent inadaptées aux besoins des utilisateurs. De plus, ces outils souffrent d'un nombre restreint de données recueillies, incapable de couvrir des volumes massifs ou variés, et présentent des insuffisances en termes d'efficacité et de conformité légale, notamment pour répondre aux exigences des réglementations. Ces lacunes renforcent la nécessité de développer une solution innovante, performante et adaptée à ces défis.

3.1.5. Description de la cible

L'application de crawling et de scraping s'adresse à une cible variée, ayant besoin d'extraire et d'organiser efficacement des informations issues du web. Les chercheurs et analystes de données ayant besoin de collecter et d'analyser des volumes importants d'informations en ligne pour leurs études et rapports. Les journalistes et rédacteurs qui cherchent à automatiser la collecte d'articles ou de contenus d'actualité pour leurs enquêtes et publications. Les entreprises et les professionnels du marketing qui souhaitent surveiller l'évolution de certains secteurs, suivre la concurrence ou recueillir des avis et tendances à partir de contenus en ligne. Les développeurs et ingénieurs en data science, impliqués dans la création de bases de données à partir de contenus en ligne, exploitant les techniques de web scraping pour l'entraînement de modèles d'intelligence artificielle. Les institutions académiques et les étudiants qui souhaitent structurer et organiser des contenus extraits pour enrichir leurs travaux et analyses.

L'outil proposé répond donc aux besoins d'un large éventail d'utilisateurs ayant besoin de collecter, organiser et analyser des informations accessibles sur le web grâce à des algorithmes avancés de crawling et de scraping.

3.1.6. Besoins Fonctionnels

Le système de crawling et de scraping repose sur plusieurs fonctionnalités essentielles pour assurer une extraction efficace et structurée des données web que ça soit la navigation automatisée et intelligente ou l'application doit être capable d'explorer automatiquement des pages web tout en s'adaptant aux différentes structures de sites, y compris ceux générés dynamiquement avec JavaScript. Le système devra récupérer uniquement les informations pertinentes définies par l'utilisateur, en filtrant les contenus inutiles tels que les publicités, les menus de navigation et les commentaires. L'algorithme intégrera des règles de classification

pour organiser les données extraites en fonction de leur pertinence et de leur type (articles, titres, images, métadonnées, etc.). Une fonctionnalité de suivi et de mise à jour automatique permettra de ré-exécuter les processus de scraping à intervalles réguliers pour capturer les nouvelles données disponibles sur les sites ciblés. Une interface utilisateur accessible permettra aux utilisateurs de configurer facilement les paramètres de crawling et de scraping, de visualiser les résultats et de les exporter sous différents formats (Excel, CSV, JSON). Le système devra respecter les fichiers robots.txt et les conditions d'utilisation des sites web scrappés afin de garantir une collecte de données légale et éthique.

Ces besoins fonctionnels garantissent un outil performant et adaptable, répondant aux exigences de collecte et d'analyse de données automatisées à grande échelle

3.1.7. Besoins Non Fonctionnels

3.1.7.1. Performance

La performance est un enjeu central du projet, car les systèmes de crawling et scraping doivent traiter de grandes quantités de données dans des délais restreints tout en optimisant les ressources utilisées. Cela implique une vitesse d'exécution élevée grâce à des algorithmes de navigation et d'extraction optimisés, une gestion efficace des ressources pour minimiser la consommation en bande passante et en puissance de calcul, et une capacité à exécuter des tâches simultanées sur plusieurs sites ou pages grâce à des approches parallèles. Pour atteindre ces objectifs, des technologies comme le framework Scrapy, des techniques d'indexation avancées, et des approches distribuées sont envisagées, garantissant ainsi un système rapide, scalable et adapté aux environnements complexes et dynamiques.

3.1.7.2. Fiabilité

La fiabilité est un pilier fondamental pour garantir que le système fonctionne de manière continue et robuste, même face aux imprévus et aux évolutions fréquentes des sites web ciblés. Le système doit démontrer une résilience face aux changements structurels, en s'adaptant dynamiquement à des modifications comme des ajustements de balises HTML ou l'ajout de captchas, grâce à l'intégration de modèles d'intelligence artificielle capables de détecter et de réagir automatiquement à ces changements. Une tolérance aux pannes est également cruciale pour limiter l'impact des erreurs ou restrictions imprévues, en permettant au système de poursuivre ses opérations globales malgré des interruptions partielles. Enfin, une journalisation des erreurs exhaustive assurera un enregistrement systématique des problèmes rencontrés, facilitant leur analyse et leur résolution rapide. Ces exigences combinées garantissent un système fiable, autonome et adaptable aux défis des environnements numériques complexes.

3.1.7.3. Sécurité

La sécurité est une priorité majeure pour ce projet, car le système interagit avec des données potentiellement sensibles et doit se protéger contre les attaques. Les informations collectées seront sécurisées grâce à un stockage protégé par chiffrement, garantissant la confidentialité des données sensibles. Des mécanismes robustes seront mis en place pour prévenir les attaques, telles que le blocage des scrapers ou les tentatives d'injection malveillante, tout en utilisant des systèmes d'authentification avancés pour restreindre les actions du scraper aux sites autorisés. En complément, des solutions techniques comme des proxys sécurisés, des réseaux VPN, et des outils anti-CAPTCHA permettront au système de naviguer de manière anonyme et protégée. Ainsi, ces mesures garantiront à la fois la sécurité des données et la résilience face aux menaces extérieures, tout en respectant les exigences éthiques et légales.

3.1.7.4. Disponibilités

Pour garantir une utilité maximale, le système doit assurer une disponibilité et un fonctionnement continu, quelles que soient les circonstances. Cela inclut un fonctionnement 24/7, permettant de traiter les données de manière ininterrompue, même en cas de charge élevée. Une surveillance en temps réel sera mise en place pour suivre les performances du système et détecter immédiatement toute défaillance, favorisant une réactivité optimale. En cas de panne, des systèmes de secours avec des mécanismes de redondance et de basculement permettront de maintenir la continuité des opérations. Enfin, le déploiement sur des plateformes cloud scalables comme AWS ou Google Cloud garantira une gestion efficace des pics de charge, tout en offrant flexibilité et robustesse au système.

3.1.7.5. Confidentialités

La confidentialité est un pilier fondamental du projet, garantissant que toutes les pratiques respectent les normes légales et éthiques. Le système devra assurer une conformité stricte avec les lois, pour protéger les droits des utilisateurs. L'anonymisation des données sera essentielle, garantissant que les informations collectées ne permettent pas d'identifier des individus ou de compromettre leur vie privée. Par ailleurs, le système respectera rigoureusement les politiques des sites web, notamment en suivant les directives des fichiers robots.txt ou autres règles spécifiques. Pour répondre à ces exigences, des protocoles de confidentialité seront intégrés, comprenant la suppression systématique des données personnelles et le respect des termes d'utilisation des sites. Ces mesures assureront une exploitation des données à la fois responsable et éthique.

3.1.7.6. Accessibilités

L'accessibilité est une priorité pour garantir que le système puisse être utilisé dans une variété de contextes technologiques et par des utilisateurs aux compétences hétérogènes. Il devra offrir

une compatibilité multiplateforme, en fonctionnant sans restriction sur des systèmes comme Windows, Linux et macOS. Une interface conviviale sera essentielle, permettant aux utilisateurs de configurer facilement des tâches de scraping ou de crawling sans nécessiter de compétences techniques avancées. Le système devra également être adaptable, avec une architecture extensible permettant d'intégrer de nouveaux modules ou technologies selon les besoins spécifiques. Pour répondre à ces objectifs, le projet intégrera des interfaces web intuitives ainsi qu'une documentation claire et détaillée pour guider les utilisateurs pas à pas, assurant ainsi une utilisation optimale et accessible à tous.

3.2. Choix des technologies

Le choix des technologies pour ce projet est guidé par leur capacité à répondre aux exigences fonctionnelles et non fonctionnelles identifiées, notamment en termes de performance, de fiabilité et de sécurité. Les technologies sélectionnées doivent permettre une intégration fluide des modules de crawling, de scraping et d'analyse des données, tout en garantissant une navigation intelligente, une extraction précise et une gestion efficace des volumes massifs de données. L'accent est également mis sur leur évolutivité et leur compatibilité avec des outils d'intelligence artificielle pour l'analyse sémantique et l'adaptabilité aux environnements dynamiques.

3.2.1. Langages de Programmation

Le projet repose principalement sur deux langages complémentaires pour répondre à ses besoins techniques :

- Python est choisi comme langage principal, en raison de son vaste écosystème de bibliothèques dédiées au web scraping, à l'intelligence artificielle et au traitement de données. Il offre une facilité d'apprentissage, une communauté active et une compatibilité élevée avec les outils nécessaires pour la mise en œuvre des modules de crawling et de scraping.
- JavaScript est utilisé pour interagir avec des sites web dynamiques grâce à des frameworks tels que Puppeteer, qui permettent de simuler le comportement d'un navigateur et de gérer efficacement les pages construites avec des technologies modernes comme React ou Angular.

3.2.2. Frameworks et Bibliothèques

Le projet s'appuie sur un ensemble d'outils et de technologies soigneusement sélectionnés pour répondre aux exigences liées au web scraping, au traitement des données, et à l'intelligence artificielle :

Pour le Crawling Web Scraping, Scrapy, un framework python puissant et scalable, conçu pour scraper de grandes quantités de données de manière rapide et efficace. Il est rapide et est intégré d'un gestion requêtes HTTP, et prise en charge native des tâches parallèles, ce qui le rend adapté aux projets d'envergure. Nous utiliserons Sélénium, un outil d'automatisation permettant d'interagir avec des sites web dynamiques, tels que ceux nécessitant la soumission de formulaires ou le clic sur des boutons générés en JavaScript. Il est idéal pour les sites complexes où des interactions utilisateur sont nécessaires.

Pour l'analyse et le traitement des Données, nous utiliserons Pandas est utilisé pour manipuler, organiser, et analyser les données collectées de manière structurée.

Pour nos visualisations, nous opterons Matplotlib une bibliothèque spécialisée permettant de créer des graphiques clairs et informatifs pour interpréter les résultats.

Pour le coté Intelligence Artificielle**,** la bibliothéque TensorFlow qui nous permettra la création et la formation de modèles d'apprentissage profond (Deep Learning) pour analyser et classifier les données. Et aussi Spacy qui est une bibliothèque avancée pour le traitement du langage naturel, particulièrement utile pour extraire du sens et des relations dans les contenus textuels collectés.

Bases de Données

On utilise PostgreSQL comme base de données relationnelle, en raison de ses performances élevées, de sa fiabilité et de sa capacité à gérer de grandes quantités de données structurées. Elle offre une flexibilité pour stocker des données complexes grâce à la prise en charge de types avancés, tels que le JSON, tout en garantissant l'intégrité des transactions et une évolutivité adaptée à des volumes croissants. PostgreSQL s'intègre idéalement dans l'architecture, assurant un stockage sécurisé et accessible des données collectées.

Interfaces Utilisateur

Pour l'interface, nous allons utiliser Flask pour le développement des interfaces web, offrant aux utilisateurs une plateforme conviviale pour configurer et visualiser les tâches de crawling et de scraping. Grâce à sa robustesse et à son cadre complet, Flask permet de créer des interfaces intuitives et sécurisées, tout en facilitant la gestion des interactions utilisateur et l'intégration avec les modules de traitement en arrière-plan. Cette solution garantit une expérience fluide et accessible, adaptée à des utilisateurs ayant des compétences techniques variées.

Chapitre 4 : Présentation de la proposition et tests

4.1. Présentation de la proposition

Notre projet a pour objectif de créer une application facile à utiliser et efficace qui aide les utilisateurs à explorer et extraire des données depuis internet. En somme, notre application permet de rendre les recherches plus simples, surtout pour les professionnels qui cherchent des infos précises. Avec notre application, on veut offrir une solution pratique pour collecter des données et les organiser de manière exploitable.

Imaginez quelqu'un qui veut chercher des informations sur l'histoire du Sénégal, comme des articles sur les grandes figures politiques ou religieuses, ou encore des événements marquants. Aujourd'hui, il faudrait passer des heures à fouiller des dizaines de pages web pour tout trouver. Avec cette application, tout ce travail est automatisé : elle explore le site, récupère les articles pertinents, et tout est prêt à être utilisé directement. Par exemple, si un étudiant en histoire sénégalaise veut rassembler des articles sur les luttes anti-coloniales, il entre simplement l'URL d'un site d'actualités sénégalaises, et l'application fait toute la collecte.

Ce qui rend notre application différente, c'est l'intégration des technologies avancées comme le NLP et le Machine Learning. Ces outils permettent de classer et de filtrer automatiquement les contenus pour ne garder que ce qui est vraiment important. Par rapport aux outils classiques, ça évite de perdre du temps avec des résultats hors sujet ou inutiles.

4.2. Architecture générale

L'architecture du système repose sur plusieurs composantes clés qui interagissent de manière harmonieuse pour assurer la collecte et l'exploitation des données historiques. Le module de crawling, basé sur l'apprentissage automatique, explore les liens pertinents de manière intelligente et efficace. Le module de scraping extrait les informations spécifiques nécessaires avec précision. Une base de données centralise et stocke les données collectées sous des formats standardisés, facilitant leur consultation et leur exploitation. Une interface utilisateur intuitive permet de configurer facilement les requêtes et de visualiser les résultats. Enfin, des composants d'intelligence artificielle enrichissent le système grâce à l'analyse sémantique et à la priorisation des pages, garantissant une collecte pertinente et optimisée.

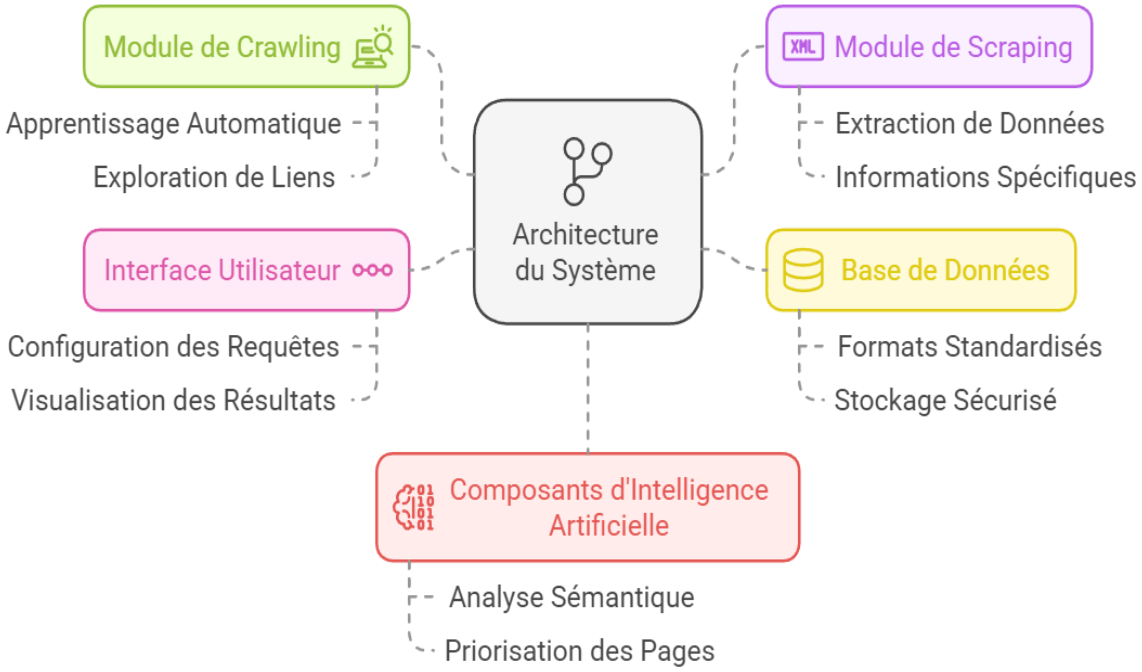


Figure 7: Architecture du système

Notre système comprend les modules suivants :

4.2.1. Le Module de crawling

Le module de crawling est la première étape du processus. Son rôle est d'explorer automatiquement les sites web, de collecter les liens internes et de les stocker pour une analyse ultérieure. L'utilisateur entre une URL ainsi qu'une profondeur de recherche, et notre crawler va suivre tous les liens jusqu'au niveau spécifié. Lorsqu'on lance le crawling, notre système envoie une requête HTTP à l'URL fournie par l'utilisateur. Ensuite, grâce aux bibliothèques comme Requests et BeautifulSoup, il récupère le contenu HTML de la page et identifie tous les liens présents. Il suit ensuite ces liens et continue l'exploration selon la profondeur définie.

Pour éviter de surcharger les serveurs, un délai entre chaque requête est appliqué. Le crawler évite de revisiter plusieurs fois la même page, grâce à un système de stockage structuré qui garde une trace des liens explorés. Tous les liens collectés sont stockés dans une base PostgreSQL pour être réutilisés lors de l'étape de scraping. Chaque lien enregistré est accompagné de sa profondeur, son temps de réponse, et sa taille pour permettre un suivi plus précis. Ce module est essentiel car il permet de créer une cartographie des pages web avant d'extraire le contenu pertinent. Sans un bon crawling, le scraping ne serait pas efficace. Maintenant qu'on a tous les liens stockés, on peut passer au module de scraping qui va extraire les informations des pages collectées.

4.2.2. Le Module de scraping :

Le module de scraping intervient juste après le crawling. Une fois que notre crawler a collecté et stocké les liens des pages web, le scraper va analyser et extraire les informations pertinentes de ces pages. L'objectif ici est de récupérer du contenu structuré, exploitable pour l'utilisateur. Lorsque l'utilisateur lance le scraping, notre système visite chaque lien enregistré dans la base de données et récupère son contenu. À partir du HTML de la page, il extrait uniquement les éléments nécessaires en se basant sur les balises définies (titres, paragraphes, dates, auteurs, etc.).

Nous utilisons BeautifulSoup pour naviguer dans le HTML et isoler les parties intéressantes du texte. Mais le scraping ne se limite pas seulement à extraire du contenu, il faut aussi nettoyer les données. C'est pourquoi notre application applique plusieurs étapes de prétraitement telle que la suppression des éléments inutiles, les menus, les publicités, les sections de commentaires, etc. Et assure le formatage du texte c'est-à-dire la conversion en encodage standard, suppression des caractères spéciaux et mise en forme homogène. L'un des points forts de notre scraper, c'est qu'il ne récupère pas tout le contenu d'une page, mais seulement ce qui est pertinent. Pour cela, nous avons intégré des techniques de traitement automatique du langage naturel (NLP) qui permettent de filtrer les articles en fonction de leur thématique.

Nous utilisons **s**paCy et le modèle facebook/bart-large-mnli pour classifier les textes et garder uniquement ceux qui correspondent aux besoins de l'utilisateur. Cela évite de récupérer des pages hors sujet et optimise la pertinence des résultats. Le module de scraping est un élément clé de l'application, car il transforme les pages brutes en données exploitables. Grâce à l'intégration du NLP, il assure un tri intelligent des informations et garantit la pertinence des résultats.

4.2.3. La Base de données :

La base de données joue un rôle essentiel dans notre application. Elle sert de stockage centralisé pour conserver toutes les informations collectées lors des différentes étapes du processus, que ce soit les liens explorés par le crawler, les contenus extraits par le scraper ou encore les comptes utilisateurs et leurs préférences. Nous avons opté pour PostgreSQL, un système de gestion de bases de données relationnelles robuste, sécurisé et performant. PostgreSQL est bien adapté à notre projet, car il permet de gérer efficacement de grandes quantités de données et offre une

gestion avancée des transactions, ce qui est essentiel lorsque l'on traite un volume important de pages web. Notre base est structurée en plusieurs tables principales :

1. La table users, qui stocke les informations des utilisateurs enregistrés, incluant leur nom, prénom, email et mot de passe haché pour garantir la sécurité.
2. La table link qui contient tous les liens collectés par le crawler avec des métadonnées comme la date d'exploration, la profondeur et l'état de traitement (scrapé ou non).
3. La table data qui stocke le contenu des pages après le scraping, avec des informations comme le titre, l'auteur, la date de publication et le texte nettoyé.

Nous avons choisi PostgreSQL pour plusieurs raisons parce qu'il gère efficacement les requêtes complexes et peut traiter un grand nombre de données sans ralentissement. Les mots de passe sont hachés et les accès sont protégés via des permissions strictes. PostgreSQL est robuste face aux crashs et assure l'intégrité des données grâce à son mécanisme de transactions ACID. L'application communique avec la base de données via SQLAlchemy, un ORM (Object Relational Mapper) qui simplifie les interactions en permettant de manipuler les données directement en Python. Cela nous permet de créer automatiquement les tables si elles n'existent pas encore. D'ajouter, modifier et supprimer des enregistrements sans devoir écrire directement du SQL brut. Et d'optimiser les requêtes pour récupérer rapidement les résultats sans surcharge inutile.

4.2.4. Une interface utilisateur

L'interface utilisateur (UI) est un élément clé de notre plateforme, car elle permet à l'utilisateur d'interagir avec les différentes fonctionnalités de l'application sans avoir besoin de connaissances techniques en programmation. Elle a été conçue de manière simple, intuitive et ergonomique pour faciliter la navigation et optimiser l'expérience utilisateur. Notre interface est développée avec Flask pour la partie backend et HTML, CSS, et JavaScript pour la partie frontend. Flask nous permet de gérer la logique métier et les requêtes entre l'utilisateur et la base de données, tandis que les langages web assurent une présentation fluide et dynamique. Nous avons aussi intégré Bootstrap pour améliorer le design et garantir une compatibilité avec les différents appareils (PC, tablette, mobile).

Notre page d'accueil présente brièvement le projet et ses fonctionnalités. Avec deux boutons principaux : "Commencer" et "Connexion".

La page de connexion et d'inscription, ou l'utilisateur peut se connecter avec son email et son mot de passe. La possibilité de créer un compte en remplissant un formulaire avec son nom, prénom, email et mot de passe. Toutes ces informations sont stockées en PostgreSQL, avec un hachage sécurisé du mot de passe.

L'espace utilisateur après connexion) est composée d'un bouton "CrawlerScrapYourSite" où l'utilisateur entre une URL et une profondeur pour lancer un crawling ciblé et un autre bouton "Full Crawler" ou l'utilisateur spécifie un mot-clé et une profondeur pour explorer des sites préconfigurés. Dans les deux cas nous avons :

- L'affichage des résultats sous forme de liste avec des options de téléchargement des fichiers Excel.
- Le suivi du crawling en temps réel avec un affichage dynamique des liens explorés.
- L'utilisateur peut activer le scraping et récupérer les données extraites.
- Une barre de progression permet de suivre l'évolution des opérations.
- Une fois le crawling ou le scraping terminé, l'utilisateur peut télécharger les résultats au format Excel ou JSON.

Nous avons mis un accent particulier sur une navigation fluide et intuitive, toutes les fonctionnalités sont accessibles en quelques clics. Un design minimaliste et efficace, pas de surcharge visuelle, juste l'essentiel pour guider l'utilisateur. Une compatibilité avec tous les navigateurs : Testé sur Chrome, Firefox, Edge et Safari. Une gestion des erreurs améliorée : Si un lien n'est pas valide ou qu'une erreur survient, un message clair est affiché pour guider l'utilisateur. L'interface communique avec le backend via des requêtes HTTP (GET et POST). Lorsqu'un utilisateur entre un lien et lance le crawling, une requête est envoyée à Flask, qui traite la demande et renvoie les résultats sous forme de JSON, ensuite affichés dynamiquement sur la page.

4.2.5. Module fonctionnel basé sur IA :

Contrairement à un simple crawler qui récupère aveuglément toutes les données d'une page web, ce module permet une analyse intelligente du contenu pour ne garder que les informations les plus pertinentes en fonction des besoins de l'utilisateur. Pour mettre en place ce module, nous avons intégré plusieurs techniques issues du Machine Learning et du Traitement Automatique du Langage Naturel :

- Pour les modèles NLP pour le filtrage des données, nous avons utilisé spaCy et NLTK pour le prétraitement des textes (suppression des stopwords, tokenization, lemmatisation), transformers (BERT, BART-large-mnli) pour la classification et la compréhension sémantique des textes et pour la détection des contenus pertinents en fonction des mots-clés définis par l'utilisateur.
- Pour l'apprentissage par renforcement (Q-Learning) pour un scraping intelligent, le système apprend à prioriser certains types de pages en fonction des résultats obtenus précédemment. Il optimise les parcours de crawling pour éviter les contenus non pertinents (publicités, sections de commentaires inutiles, etc.).
- Pour l'analyse de similarité entre les pages web, nous avons utilisé Sentence Transformers (comme SBERT) pour comparer le contenu des pages et éviter les doublons.

Notre plateforme a un impact considérable sur le Gain de temps de l'utilisateur, l'IA évite de récupérer des pages inutiles et se concentre sur l'essentiel. Nous avons une nette amélioration de la pertinence, le filtrage avancé garantit une extraction qualitative et pas seulement quantitative. Une optimisation des performances : Grâce au Q-Learning, le scraping devient plus intelligent au fil des utilisations. Et L'utilisateur obtient des données nettoyées, classées et résumées, prêtes à être exploitées. Ce module basé sur l'IA transforme un simple scraping en un processus intelligent et optimisé. Il permet de mieux structurer, analyser et exploiter les données web en intégrant les dernières avancées en NLP et Machine Learning.

4.3. Test de la plateforme

4.3.1. Page principale

L'exécution de notre application se fait en local, garantissant ainsi à l'utilisateur un contrôle total sur les données explorées et extraites. L'application repose sur Flask comme framework web et PostgreSQL pour la gestion des liens explorés et des résultats de scraping. Pour démarrer l'application, l'utilisateur exécute la commande suivante dans son terminal : **python app.py** une fois cette commande validée, le serveur Flask se met en marche et l'application est accessible via le navigateur à l'adresse : **http://127.0.0.1:5000.** L'interface d'accueil s'affiche alors, offrant à l'utilisateur une navigation fluide et intuitive.

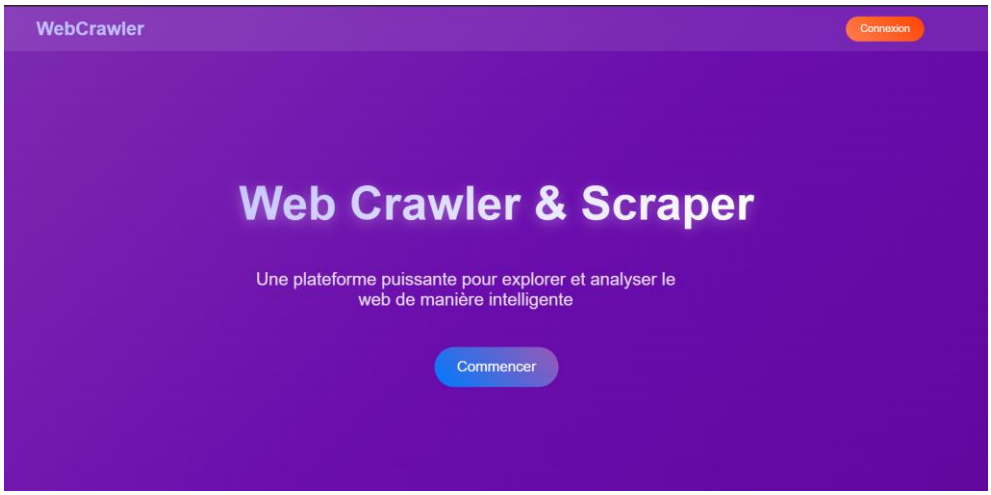


Figure 8: Page d'accueil de notre application

4.3.2. **Connexion et création de compte**

Lorsqu'un utilisateur lance l'application en local, il est accueilli par la page d'accueil, qui propose deux options principales :

Le bouton "Commencer", placé au centre de l'écran, qui permet de démarrer l'expérience utilisateur. Le bouton "Connexion", situé en haut à droite, qui permet d'accéder à l'espace utilisateur. Ces deux boutons mènent à la même interface, qui est la page de connexion.

Une fois sur cette page, l'utilisateur a deux possibilités :

- Se connecter s'il possède déjà un compte, en renseignant son email et son mot de passe avant de cliquer sur "Se connecter".
- Créer un compte s'il est nouveau sur la plateforme, en cliquant sur le lien "Créer un compte", situé en bas du formulaire de connexion.

Si l'utilisateur choisit de s'inscrire, il est redirigé vers la page d'inscription où il doit remplir les informations suivantes :

- Prénom
- Nom
- Email
- Mot de passe
- Confirmation du mot de passe

Après validation, les informations sont stockées dans une base de données PostgreSQL avec un hachage sécurisé du mot de passe pour garantir la protection des comptes.

Une fois l'inscription terminée, l'utilisateur peut retourner à la page de connexion et accéder à son espace personnel en renseignant ses identifiants.

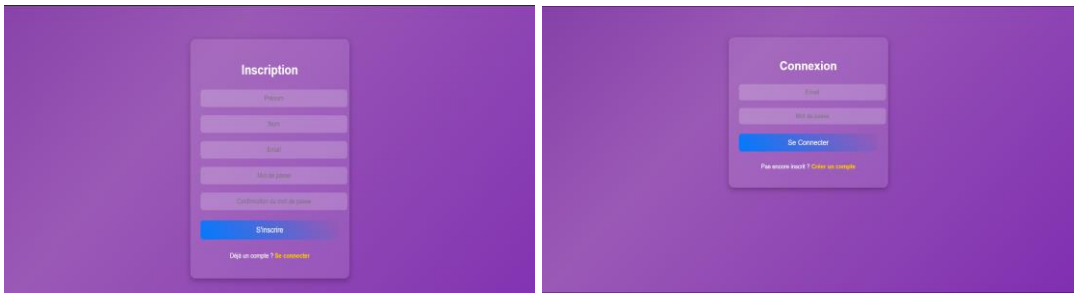


Figure 9: Page de création de compte et de connexion

4.3.3. **Page principale**

Une fois connecté, l'utilisateur est redirigé vers la page principale de l'application, où il peut accéder aux fonctionnalités de crawling et scraping via deux boutons bien distincts.

L'écran affiche un message de bienvenue "Bienvenue sur Web Crawler", suivi d'un sous-texte expliquant l'objectif principal de l'application : *"Explorez et analysez le web de manière intelligente avec nos outils avancés."*

1. "CrawlerScrapYourSite"

Cette option permet à l'utilisateur d'entrer une URL et une profondeur pour lancer un crawling ciblé sur un site spécifique. Une fois le crawling terminé, l'utilisateur pourra récupérer les liens explorés et lancer le scraping des pages pour extraire les informations pertinentes. Un fichier Excel contenant les résultats est généré automatiquement.

2. "Interface mots-clés et profondeur"

Cette option permet de lancer un crawling intelligent sans spécifier d'URL. L'utilisateur saisit un ou plusieurs mots-clés, ainsi qu'une profondeur de recherche. Le programme explore des sites préconfigurés en arrière-plan et récupère les pages contenant les termes recherchés.

L'utilisateur peut ensuite télécharger les résultats ou lancer le scraping sur ces liens filtrés.

4.3.4. Le Module CrawlerScrapYourSite

4.3.4.1. Crawling

Le module de crawling de **CrawlerScrapYourSite** est chargé d'explorer les sites web. Grâce à des bibliothèques comme requests et BeautifulSoup, l'application peut visiter une page web, identifier les liens internes, et les suivre pour découvrir de nouvelles pages.

Une profondeur maximale est définie par l'utilisateur pour éviter de surcharger les serveurs et pour limiter les explorations inutiles. Par exemple, si on choisit une profondeur de 3, le crawler explorera les liens jusqu'à trois niveaux à partir de la page de départ.

Pour garantir une exploration propre, une logique d'exclusion est mise en place. Certains types de liens, comme ceux pointant vers des commentaires, des publicités ou des sections inutiles, sont ignorés.

Les liens explorés sont ensuite stockés dans une structure organisée, ce qui permet de garder une trace des pages visitées et d'éviter les duplications.

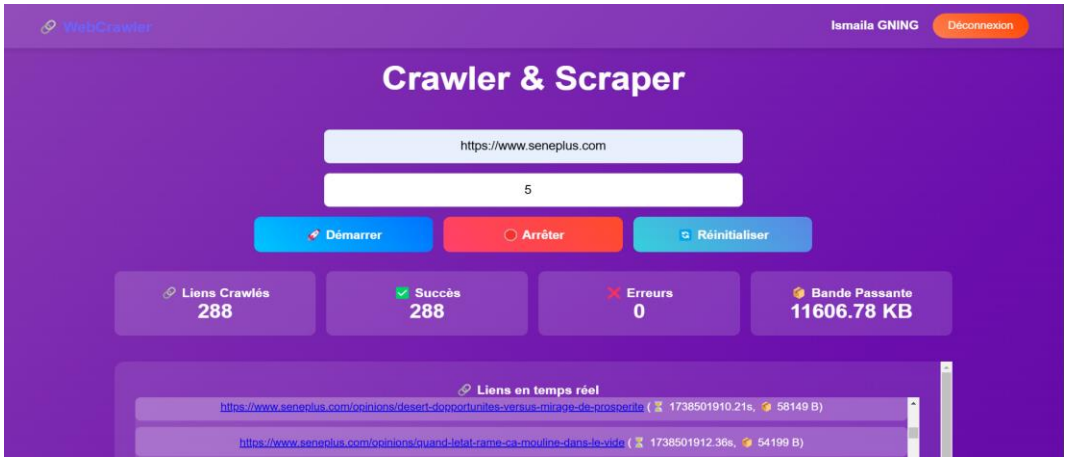


Figure 10: Page de crawling et de scraping

4.3.4.2. Scraping

Pour le scraping dans **CrawlerScrapYourSite**, c'est l'étape où on ne se contente plus de collecter des liens, mais où on extrait le contenu des pages. Ici, on utilise des techniques avancées de traitement du langage naturel pour analyser les textes récupérés.

Pour cela, des outils comme spaCy et les modèles transformers par exemple facebook/bart-large-mnli sont intégrés. Ces technologies permettent de détecter les contenus réellement pertinents. Si un article parle de l'histoire du Sénégal ou de la politique, il sera conservé, tandis que d'autres articles moins pertinents seront rejetés.

Le NLP ne se limite pas à classifier les articles. Il nettoie également le texte en supprimant les parties inutiles, comme les menus ou les publicités, pour ne garder que l'essentiel. Au final, chaque article est structuré avec un titre, une date, un auteur (si disponible), et le corps du texte.

Figure 11: Boutons d'exécution et d'arrêt du scraping

4.3.5. Le Module Interface mots-clés et profondeur

L'interface "**Mots-Clés et Profondeur**" est conçue pour permettre aux utilisateurs d'effectuer un crawling et un scraping web ciblé, basé sur des mots-clés spécifiques. Cette approche repose sur l'exploration automatisée de sites web pour extraire uniquement les contenus pertinents en fonction du mot-clé fourni. L'objectif est d'optimiser la collecte d'informations et de filtrer les données inutiles en se concentrant sur les pages en rapport avec la requête de l'utilisateur.

4.3.5.1. Description Fonctionnelle

L'interface permet à l'utilisateur de spécifier :

- Un mot-clé qui définit le sujet d'intérêt et oriente le processus de collecte des données.
- Une profondeur d'exploration un paramètre détermine jusqu'à quel niveau l'algorithme doit suivre les liens internes des pages explorées.

Une fois les paramètres définis, le crawler parcourt des sites web prédéfinis et identifie les pages dont le contenu est en lien avec le mot-clé. L'utilisateur peut suivre l'état du processus en temps réel et, une fois le crawling terminé, lancer le scraping pour extraire et analyser les informations des pages sélectionnées.

4.3.5.2. Description Technique

L'interface repose sur une architecture modulaire intégrant des techniques de crawling, scraping et traitement automatique du langage naturel pour améliorer la pertinence des résultats.

Le crawler explore des sites web prédéfinis en suivant une approche récursive :

1. Envoi d'une requête HTTP à l'URL de départ et récupération du contenu HTML.
2. Analyse du contenu de la page pour vérifier si le texte contient le mot-clé.
3. Extraction des liens internes et suivi des liens pertinents jusqu'à atteindre la profondeur maximale définie par l'utilisateur.
4. Filtrage des résultats, seules les pages contenant le mot-clé sont enregistrées.

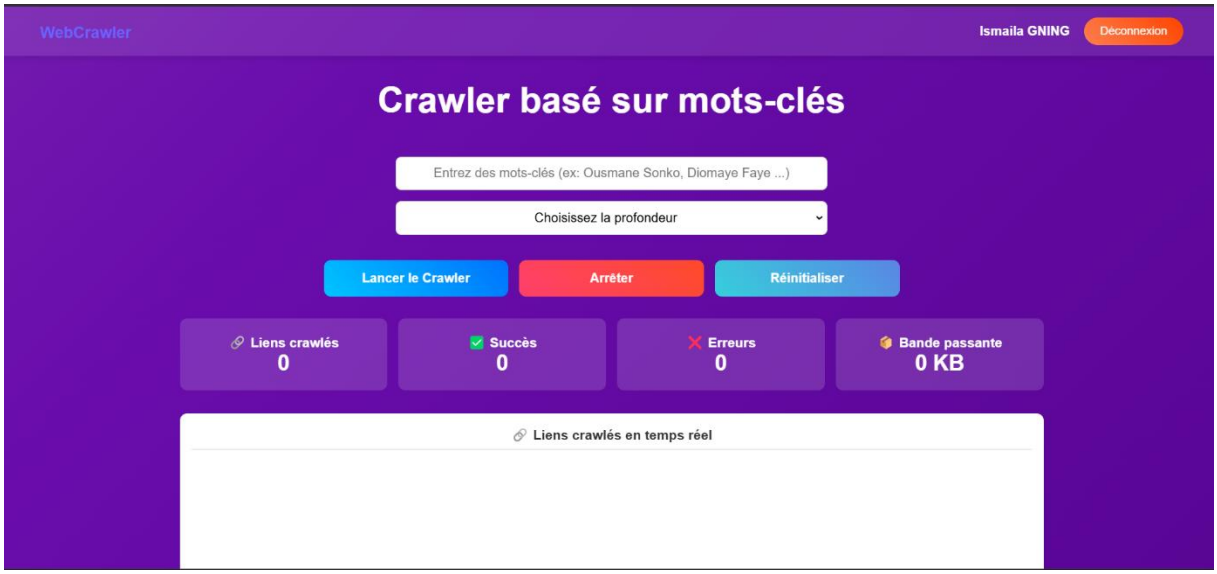


Figure 12: Interface de Crawling et Scraping par Mots-Clés

4.3.5.3. Processus de Scraping et Traitement NLP

Une fois le crawling terminé, l'utilisateur peut lancer le scraping pour extraire des informations structurées, le titre et la description de chaque page sont récupérés via des balises HTML (<title> et <meta>). Le modèle NLP, spacy est utilisé pour identifier les mots-clés dominants, effectuer une lemmatisation et extraire des résumés automatiques. Les données sont formatées dans un tableau et peuvent être exportées au format Excel.

4.3.6. Machine Learning et Optimisation

Le scraping dans nos deux interfaces, peut être optimisé grâce au Q-Learning, une méthode d'apprentissage par renforcement. L'idée est que le système apprend au fur et à mesure à choisir les meilleures stratégies pour explorer et extraire les données. Par exemple, si un type de balise ou de structure de page donne souvent de bons résultats, l'algorithme va privilégier ces options. Cette approche réduit les erreurs, améliore la qualité des articles extraits, et optimise le temps d'exécution. Avec le Q-Learning, le scraping devient plus intelligent à chaque utilisation.

4.3.7. Téléchargement et Exportation

Après le crawling ou le scraping, les utilisateurs ont besoin d'accéder aux résultats. Pour cela, l'application génère des fichiers Excel contenant toutes les informations collectées.

Si l'utilisateur veut uniquement les liens explorés, il peut télécharger un fichier listant ces derniers. S'il veut les articles extraits, un autre fichier Excel détaillé est généré, avec des colonnes pour le titre, la date, l'auteur, le texte, etc. Tout se fait directement depuis l'interface avec un simple clic.

4.3.8. Observations et Analyse l'interface

Les interfaces partagées illustrent les différentes étapes et fonctionnalités de l'application de crawling et scraping. L'interface affiche un tableau de bord des résultats obtenus après le crawling des sites web. On remarque plusieurs indicateurs essentiels pour suivre l'activité du crawler :

- Nous avons les liens explorés, qui est le nombre total de liens explorés est de 288. Ce chiffre reflète la capacité du crawler à couvrir un large éventail de contenus à partir de l'URL donnée.
- Nous avons le taux de succès, tous les liens explorés sont enregistrés comme des réussites, ce qui montre une exécution sans erreurs pour cette session.
- La bande passante consommée, la valeur affichée, 11606.78 KB, indique que l'application calcule la quantité de données téléchargées, un point important pour surveiller la performance et l'efficacité.
- Et la liste des liens en temps réel, les URLs crawlés sont affichées avec des informations supplémentaires, telles que la taille des données extraites et le temps de réponse. Cette liste permet de visualiser en direct les progrès du crawling.

Notre application développée s'inscrit dans une démarche d'optimisation de la recherche et de la centralisation des informations. En combinant crawling, scraping et exportation, elle offre aux utilisateurs une solution simple et fluide pour explorer et collecter des articles pertinents sans effort manuel.

Grâce à des fonctionnalités avancées comme PostgreSQL, qui permet de stocker et organiser efficacement les liens explorés, le traitement du langage naturel qui assure une classification et un filtrage intelligent des contenus en fonction des mots-clés définis, l'apprentissage par renforcement (Q-Learning), qui améliore progressivement la capacité de l'algorithme à identifier les articles les plus pertinents.

L'application garantit une extraction rapide, précise et automatisé**e** des données. L'approche adoptée ici pourrait être étendue à d'autres domaines et thématiques, ouvrant ainsi la voie à une utilisation plus large du scraping et du machine learning dans l'exploitation des données en ligne.

Conclusion et perspectives

Dans ce mémoire, nous avons travaillé sur la conception et la mise en place d'un crawler et scraper intelligents large spectre. L'objectif principal était de développer une application performante et automatisée permettant d'explorer le web et d'extraire des informations pertinentes en fonction des besoins des utilisateurs. Face à l'immensité des données disponibles en ligne et aux défis liés à leur récupération, nous avons cherché à proposer une solution robuste et optimisée, combinant crawling, scraping et traitement du langage naturel (NLP).

Pour atteindre cet objectif, nous avons suivi une approche méthodologique structurée, en nous basant sur :

- Une étude approfondie du web crawling et du web scraping, en mettant en évidence leurs différences et complémentarités.
- L'état de l'art des techniques avancées utilisées dans l'extraction de données, notamment le NLP et l'apprentissage automatique (Q-Learning) pour affiner la pertinence des résultats.
- La mise en place d'une architecture modulaire, intégrant Flask pour l'interface utilisateur, PostgreSQL pour le stockage des données, et des algorithmes de scraping intelligents pour une extraction optimisée.
- Une expérimentation sur des sites sénégalais, illustrant des cas d'utilisation concrets, notamment dans la collecte d'articles historiques sur le Sénégal.
- Les résultats obtenus montrent que notre solution permet une exploration efficace et automatisée du web, tout en garantissant une extraction pertinente des contenus ciblés. Grâce à l'intégration de mots-clés spécifiques et de techniques avancées de NLP, notre application offre une meilleure classification des articles, réduisant ainsi le bruit informationnel.

Notre travail ouvre plusieurs pistes d'amélioration et d'extension :

- L'optimisation du modèle d'exploration qui peut être améliorer la gestion des liens en utilisant des algorithmes de priorisation basés sur le machine learning, afin d'explorer en priorité les pages les plus pertinentes.
- La détection et contournement des blocages pour mettre en place des stratégies avancées (rotation d'IP, gestion des CAPTCHA) pour éviter les restrictions imposées par certains sites.

- Nous souhaiterons étendre le projet en intégrant un module de reconnaissance sémantique permettant de structurer automatiquement les informations collectées dans une base de connaissances accessible aux chercheurs et historiens, et aussi adapter l'algorithme pour supporter plusieurs langues locales (Wolof, Pulaar, Sérère, etc.), afin d'améliorer l'accessibilité aux ressources culturelles et historiques du Sénégal.
- Dans nos perspectives, nous souhaiterons intégrer une API pour l'accès aux données, pour permettre à d'autres applications de consommer les données extraites via une API REST, facilitant ainsi leur réutilisation par des chercheurs ou institutions.

En somme, ce travail pose les bases solides d'une application de crawling et scraping intelligente, avec des perspectives d'évolution vers un véritable outil d'analyse et de structuration des données web, en particulier pour la mise en valeur du patrimoine historique et culturel du Sénégal.

Références

[1] A. Chen et al. Improving online search accuracy using machine learning. Journal of Web Science, 2023.

[2] B. Kumar and C. Singh. Neural networks for intelligent crawlers. International Journal of AI Research, 2022.

[3] Manwar, A., & Mahalle, H. S. (2012). A VECTOR SPACE MODEL FOR INFORMATION RETRIEVAL: A MATLAB APPROACH.

[4] M. Faheem and P. Senellart. Crawl intelligent et adaptatif d'applications web pour l'archivage du web. ISI, 19(4), 2014.

[5] Pinel-Sauvagnat, K. (2004). XFIRM : un Modèle Flexible de Recherche d'Information pour le stockage et l'interrogation de documents XML.

[6] Chagheri, S., Roussey, C., Calabretto, S., & Dumoulin, C. (2012). Classification de documents combinant la structure et le contenu.

[7] Abdou, S., & Savoy, J. (2007). Considérations sur l'évaluation de la robustesse en recherche d'information.

[8] Wong, S., & Yao, Y. (1995). On modeling information retrieval with probabilistic inference.

[9] Manwar, A., & Mahalle, H. S. (2012). A VECTOR SPACE MODEL FOR INFORMATION RETRIEVAL: A MATLAB APPROACH.

[10] Hiemstra, D. (1998). A Linguistically Motivated Probabilistic Model of Information Retrieval.

[11] Abdou, S., Ruch, P., & Savoy, J. (2006). Dépister efficacement de l'information dans une banque documentaire : L'exemple de MEDLINE.

[12] Samarbakhsh, L. (2008). Web Search Algorithms and PageRank.

[13] Roul, R., Sahoo, J. K., & Arora, K. (2018). Query-Optimized PageRank: A Novel Approach.

[14] Henzinger, M. (2016). PageRank Algorithm.

[15] Yong-ping, Z. (2010). Theme-Drift of PageRank Algorithm Research.

[16] Fiala, D. (2012). Time-aware PageRank for bibliographic networks.

[17] Marx, V. (2013). Biology: The big challenges of big data. *Nature*, 498, 255-260.

[18] Soussi, R., et al. (2008). Un système d'aide à la recherche d'information en ligne basé sur les ontologies.

[19] Lejeune, G., et al. (2013). DAnIEL, parsimonious yet high-coverage multilingual epidemic surveillance.

[20] Gracia, J., et al. (2012). Challenges for the multilingual Web of Data. *J. Web Semant.*, 11, 63-71.

[21] Di Cesare, K. L. (2016). Amélioration de la précision de systèmes d'extraction de relations en utilisant un filtre générique basé sur l'apprentissage statistique.

[22] N'techobo, P. A. (2016). Annotations sémantiques et analyse de surface pour l'extraction de graphes d'abstraction de débats politiques.

[23] Xain, A., et al. (2020). Multilinguistic approach towards Information Retrieval System for Big Data. *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, 159-164.

[24] https://www.cdp.sn/supports-vulgarisation/booklet-loi-2008-12/Loi_2008-12_[booklet].pdf

[25] Segev, A., & Gal, A. (2008). Enhancing portability with multilingual ontology-based knowledge management. *Decis. Support Syst.*, 45, 567-584.

[26] Adomavicius, G., & Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6), 734-749.

[27] Jansen, B. J., & Spink, A. (2006). How are we searching the World Wide Web? A longitudinal study of search engine queries. *Information Processing & Management*, 42(1), 248-263.

[28] European Union. (2016). General Data Protection Regulation (GDPR). Official Journal of the European Union.

[29] Kosala, R., & Blockeel, H. (2000). Web Mining Research: A Survey. *ACM SIGKDD Explorations Newsletter*, 2(1), 1-15.

[30] Ferrara, E., De Meo, P., Catanese, S., & Fiumara, G. (2014). Web Data Extraction, Applications and Techniques: A Survey. *ACM Transactions on Internet Technology*, 14(1), 1-40.

[31] European Union. (2016). General Data Protection Regulation (GDPR). Official Journal of the European Union.

[32] Cardoso, J., & Sheth, A. (2005). Semantic e-Workflow Composition. *Journal of Business Process Management*, 11(1), 3-18.

[33] Ferrara, E., De Meo, P., Catanese, S., & Fiumara, G. (2014). *Web Data Extraction, Applications and Techniques: A Survey*. ACM Transactions on Internet Technology, 14(1), 1-40.

[34] Munzert, S., Rubba, C., Meißner, P., & Nyhuis, D. (2014). *Automated Data Collection with R: A Practical Guide to Web Scraping and Text Mining*. Wiley.

[35] Ferrara, E., De Meo, P., Catanese, S., & Fiumara, G. (2014). Web Data Extraction, Applications and Techniques: A Survey. ACM Transactions on Internet Technology, 14(1), 1-40.

[36] Munzert, S., Rubba, C., Meißner, P., & Nyhuis, D. (2014). Automated Data Collection with R: A Practical Guide to Web Scraping and Text Mining. Wiley.

[37] Wang, D., & Liu, S. (2018). A Review of Automated Web Scraping Tools for Big Data. Journal of Information Science Research, 15(3), 123-136.

[38] Singh, A., & Kumar, N. (2020). Web Scraping Using Modern Tools: A Practical Approach. International Journal of Data Science Research, 5(2), 34-45.

[39] Brin, S., & Page, L. (1998). "The Anatomy of a Large-Scale Hypertextual Web Search Engine." *Computer Networks and ISDN Systems*.

[40] Chen, H., Chiang, R. H., & Storey, V. C. (2012). "Business Intelligence and Analytics: From Big Data to Big Impact." *MIS Quarterly*.

[41] Brynjolfsson, E., & Smith, M. D. (2000). "Frictionless Commerce? A Comparison of Internet and Conventional Retailers." *Management Science*.

[42] Lawrence, S., & Giles, C. L. (1998). "Searching the Web: General and Scientific Information Access." *IEEE Communications*.

[43] Thelwall, M., & Wilkinson, D. (2003). "Web Crawling and Search Engine Architecture." *Journal of Information Science*.

[44] Moore, T., & Clayton, R. (2008). "The Impact of Incentives on Notice and Take-down." *Journal of Cybersecurity*.

[45] Kahle, B. (1997). "Preserving the Internet." *Scientific American*.

[46] Liu, B. (2012). "Sentiment Analysis and Opinion Mining." *Synthesis Lectures on Human Language Technologies*.

[47] Negash, S. (2004). "Business Intelligence." *Communications of the Association for Information Systems*.

[48] Evans, D. S. (2009). "The Online Advertising Industry: Economics, Evolution, and Privacy." *Journal of Economic Perspectives*.

[49] Barba, G., Lazoi, M., & Lezzi, M. (2024). Bibliometric Insights into Web Scraping and Advanced AI-Based Models for Valuable Business Data. , 321-328.

[50] Khan, F., Tsaramirsis, G., Ullah, N., Nazmudeen, M., Jan, S., & Ahmad, A. (2020). Smart algorithmic based web crawling and scraping with template autoupdate capabilities. Concurrency and Computation: Practice and Experience, 33

[51] Barwary, M., Jacksi, K., & Al-Zebari, A. (2023). Constructing a Multilingual E-Learning Ontology through Web Crawling and Scraping. Int. J. Commun. Networks Inf. Secur., 15, 137-153.

[52] Durga, C., J, A., Bansal, S., Singh, N., Kalra, R., & Bader, N. (2024). Adaptive Web Crawling Strategies Based on Ontological User Interest Modeling for Personalized Content Retrieval. 2024 International Conference on Trends in Quantum Computing and Emerging Business Technologies, 1-5. https://doi.org/10.1109/TQCEBT59414.2024.10545060.

Table des matières

Annexe 1 : Code pour la création de compte et de connexion

```html
<!DOCTYPE html>
<html lang="fr">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Inscription</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='css/formulaire.css') }}" />
  </head>
  <body>
    <div class="container-form">
      <h2>Inscription</h2>

      {% with messages = get_flashed_messages(with_categories=true) %}
      {% if messages %}
        {% for category, message in messages %}
        <p class="{{ category }}">{{ message }}</p>
        {% endfor %}
      {% endif %}
      {% endwith %}

      <form action="{{ url_for('register') }}" method="POST">
        <input type="text" name="firstname" placeholder="Prénom" required />
        <input type="text" name="lastname" placeholder="Nom" required />
        <input type="email" name="email" placeholder="Email" required />
        <input type="password" name="password" placeholder="Mot de passe" required />
        <input type="password" name="confirm_password" placeholder="Confirmation du mot de passe" required />

        <button type="submit" class="btn-form">S'inscrire</button>
      </form>

      <p>Déjà un compte ? <a href="{{ url_for('login') }}" class="switch-form">Se connecter</a></p>
    </div>
  </body>
</html>
```

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Connexion</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='css/formulaire.css') }}">
    <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
</head>
<body>
    <div class="container-form">
        <h2>Connexion</h2>

        <!-- Messages Flash -->
        {% with messages = get_flashed_messages(with_categories=True) %}
            {% if messages %}
                <div class="flash-messages">
                    {% for category, message in messages %}
                        <div class="flash flash-{{ category }}">{{ message }}</div>
                    {% endfor %}
                </div>
            {% endif %}
        {% endwith %}

        <form action="{{ url_for('login') }}" method="POST">
            <input type="email" name="email" placeholder="Email" required>
            <input type="password" name="password" placeholder="Mot de passe" required>

            <button type="submit" class="btn-form">Se Connecter</button>
        </form>

        <p>Pas encore inscrit ? <a href="{{ url_for('register') }}" class="switch-form">Créer un compte</a></p>
    </div>

    <script>
        // Animation des messages flash (disparaissent après 3 secondes)
        $(document).ready(function() {
            setTimeout(function() {
                $(".flash").fadeOut("slow");
```

Annexe 2 : Connexion dans notre base de données Postgres

Smart_crawler > app1.py > ...

```python
from flask import Flask, render_template, request, redirect, url_for, flash, jsonify, Response  # type: ignore
from flask_sqlalchemy import SQLAlchemy  # type: ignore
from flask_login import LoginManager, UserMixin, login_user, logout_user, login_required, current_user  # type: ignore
from werkzeug.security import generate_password_hash, check_password_hash  # type: ignore
import threading
import requests  # type: ignore
from bs4 import BeautifulSoup  # type: ignore
from urllib.parse import urljoin, urlparse
import time
import json
from collections import deque
import pandas as pd  # type: ignore
from io import BytesIO, StringIO
import re
from datetime import datetime
from collections import defaultdict
import random
import csv
from transformers import pipeline  # type: ignore
import spacy  # type: ignore
import os
import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
from dateutil import parser

# ------------------------------------------------------------------------
# 📌 Configuration de l'application Flask
app = Flask(__name__)
app.config['SECRET_KEY'] = "supersecretkey"
app.config['SQLALCHEMY_DATABASE_URI'] = 'postgresql://postgres:Passer123@localhost/crawler_db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False

# ------------------------------------------------------------------------
# 📌 Initialisation de SQLAlchemy et Flask-Login
db = SQLAlchemy(app)
login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = "login"

```

Annexe 3 : Notre Crawler

```python
# 📌 Fonction de crawling
def crawl_website(base_url, max_depth=3):
    with app_lock:
        data.update({
            "links_crawled": [],
            "errors": 0,
            "bandwidth_used": 0,
            "crawling": True,
            "stop_requested": False
        })

    visited = set()
    queue = deque([(base_url, 0)])  # (URL, profondeur)
    base_domain = urlparse(base_url).netloc

    while queue and not data["stop_requested"]:
        url, depth = queue.popleft()

        if url in visited or depth > max_depth:
            continue
        visited.add(url)

        try:
            response = requests.get(url, timeout=10, headers={'User-Agent': 'Mozilla/5.0'}, verify=False)
            response_time = time.time()

            with app_lock:
                data["bandwidth_used"] += len(response.content)

            if response.status_code == 200:
                soup = BeautifulSoup(response.text, "html.parser")
                with app_lock:
                    data["links_crawled"].append({
                        "url": url,
                        "response_time": response_time,
                        "size": len(response.content)
```

Annexe 4 : Une partie de notre scrapeur

```python
# ----------------------------------------------------------------------------
# 📥 Fonction pour récupérer le contenu d'une page
def fetch_page_original(url):
    try:
        response = requests.get(url, timeout=10, verify=False)
        if response.status_code == 200:
            return BeautifulSoup(response.content, "html.parser")
    except requests.RequestException as e:
        print(f"❌ Erreur lors de la récupération {url}: {e}")
    return None

# ----------------------------------------------------------------------------
# 🎯 Filtrage des liens inutiles
def is_valid_link(url):
    return not any(pattern in url for pattern in EXCLUDE_PATTERNS)

# ----------------------------------------------------------------------------
# 🌍 Extraire le nom du site (domaine)
def get_site_name(url):
    return urlparse(url).netloc.replace("www.", "")

# ----------------------------------------------------------------------------
# 🔥 Fonction principale pour scraper les articles
def scrape_articles(input_file, output_file):
    global scraped_data  # Ajouter la référence à la variable globale
    scraped_data = []    # Réinitialiser la liste des données

    # Définir tous les champs possibles
    fieldnames = [
        "site_name", "url", "title", "article_text", "date", "author",
        "keywords", "categories", "word_count"
    ]

    with open(input_file, mode="r", encoding="utf-8") as infile, \
         open(output_file, mode="w", newline="", encoding="utf-8") as outfile:
        reader = csv.DictReader(infile)
        writer = csv.DictWriter(outfile, fieldnames=fieldnames)
```

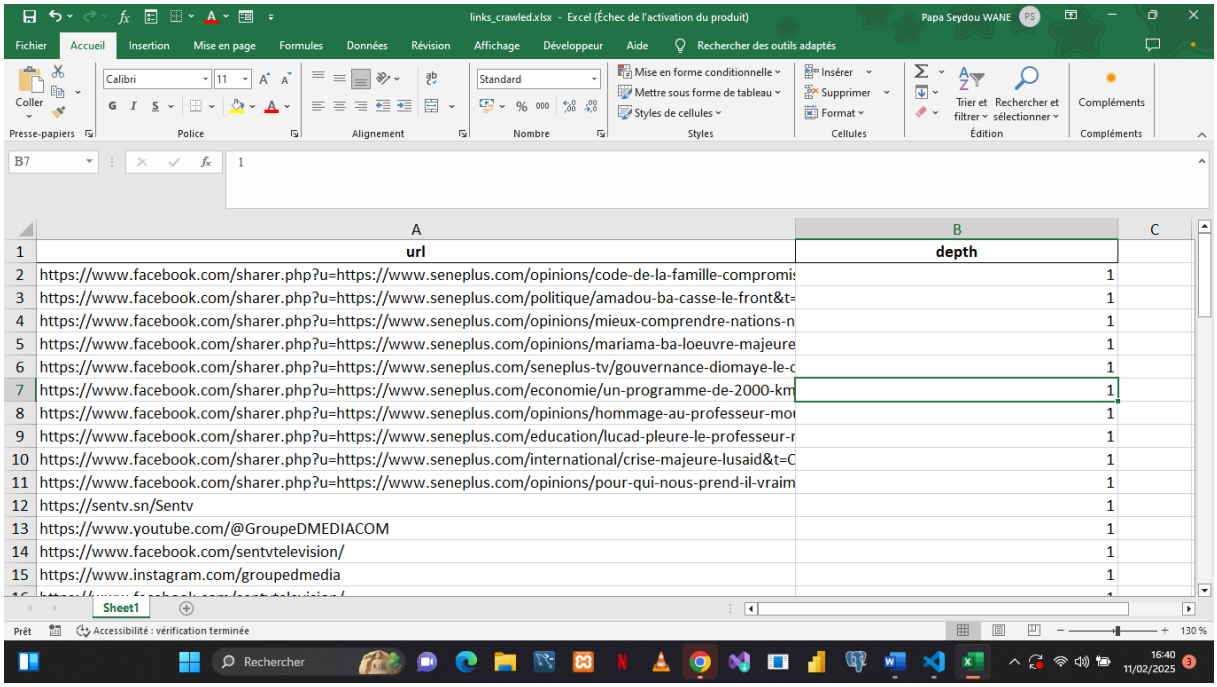
```python
    # Définir tous les champs possibles
    fieldnames = [
        "site_name", "url", "title", "article_text", "date", "author",
        "keywords", "categories", "word_count"
    ]

    with open(input_file, mode="r", encoding="utf-8") as infile, \
         open(output_file, mode="w", newline="", encoding="utf-8") as outfile:
        reader = csv.DictReader(infile)
        writer = csv.DictWriter(outfile, fieldnames=fieldnames)
        writer.writeheader()

        for row in reader:
            url = row.get("Link", "")  # Utiliser get() pour éviter les erreurs de clé
            if not url or not is_valid_link(url):
                print(f"🔸 Ignoré : {url}")
                continue

            print(f"📝 Scraping {url}...")
            data_article = extract_article_data(url)

            if data_article:
                # S'assurer que tous les champs sont présents, même vides
                row_data = {field: data_article.get(field, "") for field in fieldnames}
                writer.writerow(row_data)
                scraped_data.append(data_article)  # Ajouter à la liste en mémoire
                print(f"✅ Article extrait ({data_article.get('word_count', 0)} mots) pour {url}")
            else:
                print(f"❌ Échec de l'extraction pour {url}")

    print(f"✨ Scraping terminé. {len(scraped_data)} articles extraits.")
    return len(scraped_data)

# ----------------------------------------------------------------------------
```

Annexe 5 : Une partie de notre feuille style.css

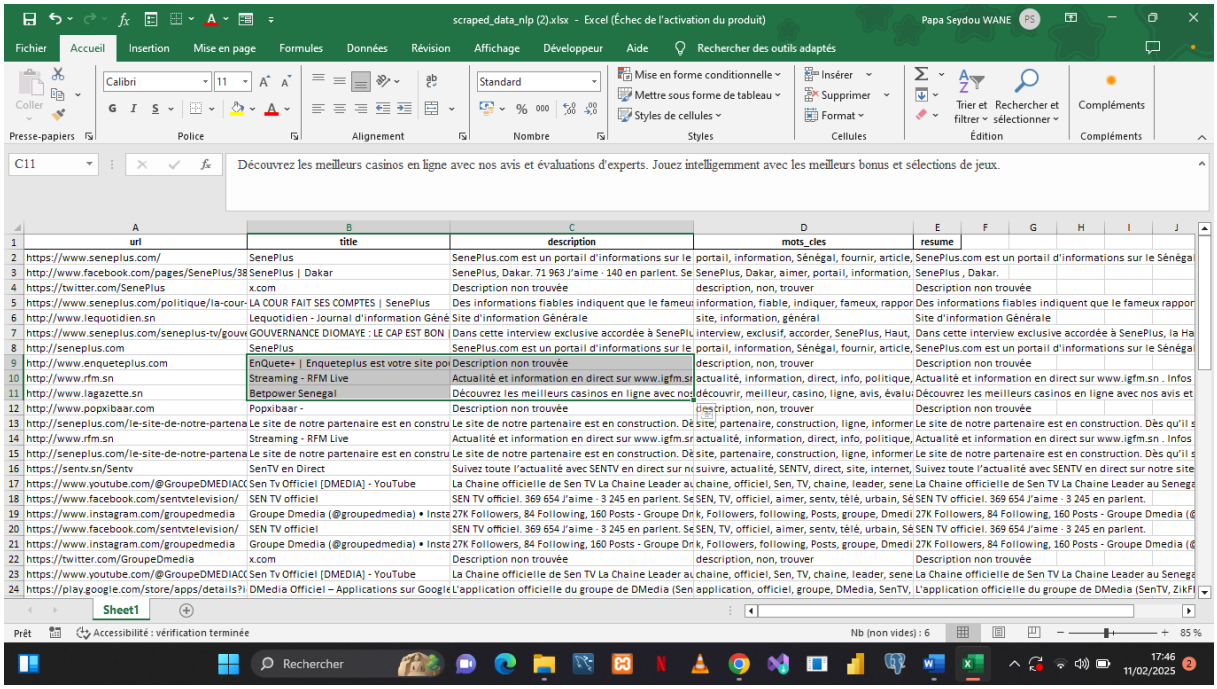
```css
/* ✅ Global */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: 'Poppins', sans-serif;
    background: linear-gradient(to right, #1e3c72, #2a5298);
    color: white;
    height: 100vh;
    display: flex;
    flex-direction: column;
}

/* ✅ Fixer le header en haut de la page */
header {
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    background: #ffffff; /* Fond blanc */
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
    z-index: 1000; /* Toujours au-dessus du reste */
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 15px 30px;
}

/* ✅ Espacement sous le header pour éviter le chevauchement */
.crawler-container {
    padding-top: 90px; /* Marge suffisante sous le header */
    text-align: center;
}

/* ✅ Style du logo cliquable */
```


```css
/* ✅ Global */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    font-family: 'Poppins', sans-serif;
    background: linear-gradient(to right, #1e3c72, #2a5298);
    color: white;
    height: 100vh;
    display: flex;
    flex-direction: column;
}

/* ✅ Fixer le header en haut de la page */
header {
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    background: #ffffff; /* Fond blanc */
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
    z-index: 1000; /* Toujours au-dessus du reste */
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 15px 30px;
}

/* ✅ Espacement sous le header pour éviter le chevauchement */
.crawler-container {
    padding-top: 90px; /* Marge suffisante sous le header */
    text-align: center;
}

/* ✅ Style du logo cliquable */
```

Annexe 6 : Feuilles Excel téléchargées après crawling et après scraping



	A	B	C
1	url	depth	
2	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/opinions/code-de-la-famille-compromis	1	
3	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/politique/amadou-ba-casse-le-front&t=	1	
4	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/opinions/mieux-comprendre-nations-n	1	
5	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/opinions/mariama-ba-loeuvre-majeure	1	
6	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/seneplus-tv/gouvernance-diomaye-le-c	1	
7	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/economie/un-programme-de-2000-km	1	
8	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/opinions/hommage-au-professeur-mou	1	
9	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/education/lucad-pleure-le-professeur-r	1	
10	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/international/crise-majeure-lusaid&t=C	1	
11	https://www.facebook.com/sharer.php?u=https://www.seneplus.com/opinions/pour-qui-nous-prend-il-vraim	1	
12	https://sentv.sn/Sentv	1	
13	https://www.youtube.com/@GroupeDMEDIACOM	1	
14	https://www.facebook.com/sentvtelevision/	1	
15	https://www.instagram.com/groupedmedia	1	



	A	B	C	D	E
1	url	title	description	mots_cles	resume
2	https://www.seneplus.com/	SenePlus	SenePlus.com est un portail d'informations sur le	portail, information, Sénégal, fournir, article,	SenePlus.com est un portail d'informations sur le Sénéga
3	http://www.facebook.com/pages/SenePlus/38	SenePlus | Dakar	SenePlus, Dakar. 71 963 J'aime · 140 en parlent. Se	SenePlus, Dakar, aimer, portail, information,	SenePlus , Dakar.
4	https://twitter.com/SenePlus	x.com	Description non trouvée	description, non, trouver	Description non trouvée
5	https://www.seneplus.com/politique/la-cour-	LA COUR FAIT SES COMPTES | SenePlus	Des informations fiables indiquent que le fameu	information, fiable, indiquer, fameux, rappor	Des informations fiables indiquent que le fameux rappor
6	http://www.lequotidien.sn	Lequotidien - Journal d'information Géné	Site d'information Générale	site, information, général	Site d'information Générale
7	https://www.seneplus.com/seneplus-tv/gouv	GOUVERNANCE DIOMAYE : LE CAP EST BON	Dans cette interview exclusive accordée à SenePl	interview, exclusif, accorder, SenePlus, Haut,	Dans cette interview exclusive accordée à SenePlus, la Ha
8	http://seneplus.com	SenePlus	SenePlus.com est un portail d'informations sur le	portail, information, Sénégal, fournir, article,	SenePlus.com est un portail d'informations sur le Sénéga
9	http://www.enqueteplus.com	ÉnQuete+ | Enqueteplus est votre site poi	Description non trouvée	description, non, trouver	Description non trouvée
10	http://www.rfm.sn	Streaming - RFM Live	Actualité et information en direct sur www.igfm.sr	actualité, information, direct, info, politique,	Actualité et information en direct sur www.igfm.sn . Infos
11	http://www.lagazette.sn	Betpower Senegal	Découvrez les meilleurs casinos en ligne avec nos	découvrir, meilleur, casino, ligne, avis, évalu	Découvrez les meilleurs casinos en ligne avec nos avis et
12	http://www.popxibaar.com	Popxibaar -	Description non trouvée	description, non, trouver	Description non trouvée
13	http://seneplus.com/le-site-de-notre-partena	Le site de notre partenaire est en constru	Le site de notre partenaire est en construction. Dè	site, partenaire, construction, ligne, informer	Le site de notre partenaire est en construction. Dès qu'il s
14	http://www.rfm.sn	Streaming - RFM Live	Actualité et information en direct sur www.igfm.sr	actualité, information, direct, info, politique,	Actualité et information en direct sur www.igfm.sn . Infos
15	http://seneplus.com/le-site-de-notre-partena	Le site de notre partenaire est en constru	Le site de notre partenaire est en construction. Dè	site, partenaire, construction, ligne, informer	Le site de notre partenaire est en construction. Dès qu'il s
16	https://sentv.sn/Sentv	SenTV en Direct	Suivez toute l'actualité avec SENTV en direct sur n	suivre, actualité, SENTV, direct, site, internet,	Suivez toute l'actualité avec SENTV en direct sur notre site
17	https://www.youtube.com/@GroupeDMEDIACO	Sen Tv Officiel [DMEDIA] - YouTube	La Chaine officielle de Sen TV La Chaine Leader au	chaine, officiel, Sen, TV, chaine, leader, sene	La Chaine officielle de Sen TV La Chaine Leader au Senega
18	https://www.facebook.com/sentvtelevision/	SEN TV officiel	SEN TV officiel. 369 654 J'aime · 3 245 en parlent. Se	SEN, TV, officiel, aimer, sentv, télé, urbain, Sé	SEN TV officiel. 369 654 J'aime · 3 245 en parlent.
19	https://www.instagram.com/groupedmedia	Groupe Dmedia (@groupedmedia) • Insta	27K Followers, 84 Following, 160 Posts - Groupe Dm	k, Followers, following, Posts, groupe, Dmedi	27K Followers, 84 Following, 160 Posts - Groupe Dmedia (@
20	https://www.facebook.com/sentvtelevision/	SEN TV officiel	SEN TV officiel. 369 654 J'aime · 3 245 en parlent. Se	SEN, TV, officiel, aimer, sentv, télé, urbain, Sé	SEN TV officiel. 369 654 J'aime · 3 245 en parlent.
21	https://www.instagram.com/groupedmedia	Groupe Dmedia (@groupedmedia) • Insta	27K Followers, 84 Following, 160 Posts - Groupe Dm	k, Followers, following, Posts, groupe, Dmedi	27K Followers, 84 Following, 160 Posts - Groupe Dmedia (@
22	https://twitter.com/GroupeDmedia	x.com	Description non trouvée	description, non, trouver	Description non trouvée
23	https://www.youtube.com/@GroupeDMEDIACO	Sen Tv Officiel [DMEDIA] - YouTube	La Chaine officielle de Sen TV La Chaine Leader au	chaine, officiel, Sen, TV, chaine, leader, sene	La Chaine officielle de Sen TV La Chaine Leader au Senega
24	https://play.google.com/store/apps/details?i	DMedia Officiel – Applications sur Google	L'application officielle du groupe de DMedia (Sen	application, officiel, groupe, DMedia, SenTV,	L'application officielle du groupe de DMedia (SenTV, ZikFl