ÉCOLE NATIONALE SUPÉRIEURE D'ARTS ET MÉTIERS
UNIVERSITÉ HASSAN II DE CASABLANCA

RAPPORT DE PROJET

Projet de classification des journaux à l'aide de L'intelligence artificielle

Juillet 2020 - Septembre 2020

Réalisé par :

- **AMRI Tarik**

Étudiante en 2ème année cycle d'ingénieur à l'ENSAM Casablanca – Intelligence Artificielle et Génie Informatique

- **BAALLA Yasser**

Étudiante en 2ème année cycle d'ingénieur à l'ENSAM Casablanca – Intelligence Artificielle et Génie Informatique

- **EL HADDAJI Khaled**

Étudiante en 2ème année cycle d'ingénieur à l'ENSAM Casablanca – Génie Electromécanique

Année scolaire 2020-2021

Plan du rapport :

I. INTRODUCTION :

Avant d'amorcer tout développement, la crise actuelle a impacté négativement les recrutements des stagiaires, la plupart ont été suspendus ou reportés. De ce fait, on a décidé de faire un projet nous permettant de mettre en pratique nos connaissances et expériences personnelles.

Ayant une passion commune pour la technologie web et l'intelligence artificielle, notre groupe composé de Tarik AMRI, Yasser BAALLA et Khalid EL HADDAJI, a vu une grande opportunité d'exploiter cet intérêt commun pour réaliser un projet personnel innovant.

II. DESCRIPTION DU PROJET

i. Technologies utilisées :

Dans les parties qui suivent, on mentionne les principales technologies utilisées durant le cycle de développement :

a - Langages Utilisés :

- HTML, CSS
- JavaScript
- Python

- MS-DOS (on convertira le code “batch” plus tard vers “bash” car le serveur sera basé sur linux alors qu’on a développé sur Windows)

b - Librairies et Frameworks principales :

Pour python :

- Django (https://www.djangoproject.com/)
- Pandas (https://pandas.pydata.org/)
- Numpy (https://numpy.org/)
- Scikit learn (https://scikit-learn.org/)
- Pickle (https://docs.python.org/3/library/pickle.html)
- Json (https://docs.python.org/fr/3/library/json.html)
- beautifulsoup4 (https://www.crummy.com/software/BeautifulSoup/bs4/doc/)
- polling2 (https://polling2.readthedocs.io/en/latest/)
- requests (https://requests.readthedocs.io/en/master/)
- regex (https://docs.python.org/3/library/re.html)
- nltk (https://www.nltk.org/)
- psycopg2 (https://www.psycopg.org/docs/)

Pour javascript :

- React (https://fr.reactjs.org/)
- Axios (https://github.com/axios/axios)
- Material-UI (https://material-ui.com/)
- MDI (https://materialdesignicons.com/)

c - Autres Technologies :

- postgresql (https://www.postgresql.org/)
- github (https://github.com/)
- google g suite (https://gsuite.google.com/intl/en/products/drive/)

III. DÉMARCHE DU PROJET :

Le projet peut être séparé en 2 parties principales, la partie intelligence artificielle et la partie web.

i - Intelligence Artificielle :

Cette partie constitue la base du projet, on parlera des principales étapes suivies ainsi que d'autres détails importants à notre création des “classifiers” qu’on va implémenter durant la partie web.
Comme la plupart des projets de machine learning, on a 3 étapes majeures à suivre : la collecte et traitement de données, puis l'entraînement de l’algorithme.

a - Collecte de Données :

On parlera durant cette partie des méthodes utilisées pour collecter nos données, que ce soit pour entraîner notre algorithme ou pour peupler notre base de données durant la partie web.

1 - Etiquetage :

Notre algorithme d’apprentissage est supervisé, cela veut dire qu’on a besoin de lui fournir des données étiquetées avant qu’il et il se basera sur ces données pour distinguer de futurs articles. Malheureusement, il n’y a pas de sources qui classifient les articles marocains. La seule forme de classification similaire est présente sur les sites de presse eux-mêmes mais elle est généralisée et imprécise pour notre besoin, il nous a fallu donc étiqueter les données nous même.
On a commencé par collecter 150 articles depuis 6 sources différentes (hespress, le360 et filiales, welovebuzz et lesEco) et on a remplis 4 colonnes pour chaque articles : le lien de l’article, les catégories, l'émotion et la complexité.
Ce travail demande un long temps pour être effectué et se fait généralement par une prestation.
Sur la durée du projet on a augmenté ce nombre à 1500 articles.

2- Web Scraping :

La première étape de la collecte qui nous vient à l'esprit c'est l'extraction des informations dont nous aurons besoin par la suite dans le traitement des données et l'entraînement de l'algorithme. C'est ici qu'intervient le Web Scraping.

Le Web Scraping est une technique permettant l'extraction des données d'un site via un programme, un logiciel automatique ou un autre site. L'objectif est donc d'extraire le contenu d'une page d'un site de façon structurée. Le scraping permet ainsi de pouvoir réutiliser ces données. Des scripts en langage python nous ont permis d'extraire le titre et le contenu de chaque article. Ces fonctions de scraping ont été implémentées pour les six différentes sources en s'appuyant sur la librairie python Beautiful Soup.

Beautiful Soup est une bibliothèque Python d'analyse syntaxique de documents HTML et XML créée par Leonard Richardson. Elle produit un arbre syntaxique qui peut être utilisé pour chercher des éléments ou les modifier. Lorsque le document HTML ou XML est mal formé (par exemple s'il manque des balises fermantes), Beautiful Soup propose une approche à base d'heuristiques afin de reconstituer l'arbre syntaxique sans générer d'erreurs. Cette approche est aussi utilisée par les navigateurs web modernes.

Bien évidemment, nous utiliserons ces données à bon escient avec un seul but en tête : classifier l'actualité nationale ainsi qu'internationale en différentes catégories et permettre un accès à l'information présentée par différentes sources.

3 - Polling :

Maintenant que nous sommes arrivés à extraire les données voulues, nous devons trouver un moyen pour envoyer des requêtes continues et répétées aux différentes sources pour savoir s'il y a de nouveaux articles ou pas. La solution s'avère la technique de polling. Le polling est une technique de programmation que les processus utilisent lorsqu'ils vérifient de façon répétée si une condition est vraie. Nous avons réalisé ainsi une fonction de polling pour chaque source. Celles-ci envoient

chaque minute des requêtes pour savoir s'il y a de nouveaux articles ou pas.

4 - Stockage :

Bien sur on va devoir stocker toutes les données qu'on collecte, et puisqu'on a un nombre assez important de données, il faudra aussi utilisé un système efficace afin d'éviter les ralentissements et les longues périodes d'attentes mais chaque cas a besoin de fonctionnalités différentes donc on a utilisé 4 solutions de stockages durant les différentes étapes : la base de données postgres, le "pickle serializer", le "json serializer" et les fichiers texte encodés en UTF-8 (txt et csv).

On a utilisé la base de données postgres durant la partie web depuis django avec la librairie "psycopg2" donc on reparlera de cette solution dans la partie III-ii-b (base de données).

"En informatique, la sérialisation (de l'anglais américain serialization) est le codage d'une information sous la forme d'une suite d'informations plus petites (dites atomiques) pour, par exemple, sa sauvegarde ou son transport sur le réseau. L'activité réciproque, visant à décoder cette suite pour créer une copie conforme de l'information d'origine, s'appelle la désérialisation"
Le pickle serializer et le json serializer effectuent tout les deux la sérialisation, mais avec une manière différente et pour des buts différents.

Le json serializer possède un format similaire aux objets du langage javascript, on l'utilise donc pour sérialiser des objets et des listes de python afin de les stocker ou les transférer vers notre front end.

Le pickle serializer en contrepartie code les données sous forme binaire, donc on pourra stocker des données plus compliquées comme un "vectorizer" ou "classifier". On parlera plus du pickle serializer après avoir introduit et défini ces types de données.

Les fichiers textes et csv sont utilisé principalement parce qu'ils sont faciles à utiliser et peuvent être ouverts et modifié par une multitude de logiciels (bloc note, excel) donc ils sont principalement utilisé lorsqu'il ya une interaction directe avec nous en tant qu'humains (pour vérifier, tester ou ajouter des données manuellement).

b - Traitement de Données :

Même avec toutes ces données, notre machine ne pourra pas simplement les comprendre.
Il faudra donc traiter ces données afin qu'on puisse extraire le maximum de données utilisable puis les rendre compréhensible à la machine.
On explorera ces techniques dans les parties qui suivent.

1 - Natural Language Processing (NLP) :

La NLP ou **traitement automatique du langage naturel** est un domaine multidisciplinaire impliquant la linguistique, l'informatique et l'intelligence artificielle, qui vise à créer des outils de traitement de la langue naturelle pour diverses applications. Il ne doit pas être confondu avec la linguistique informatique, qui vise à comprendre les langues au moyen d'outils informatiques.
L'objectif ultime de la NLP est de lire, déchiffrer, comprendre et donner un sens aux langues humaines d'une manière précieuse.
La plupart des techniques de NLP reposent sur l'apprentissage automatique pour tirer un sens des langues humaines.
En fait, une interaction typique entre les humains et les machines utilisant le NLP pourrait se dérouler comme suit:

- Un humain parle à la machine
- La machine capture l'audio
- La conversion audio-texte
- Traitement des données du texte
- La conversion des données en audio
- La machine répond à l'humain en lisant le fichier audio

À quoi sert la NLP?

La NLP est le moteur des applications courantes suivantes:

- Applications de traduction de langue telles que Google Translate
- Processeurs de texte tels que Microsoft Word et Grammarly qui utilisent la PNL pour vérifier l'exactitude grammaticale des textes.
- Applications de réponse vocale interactive (IVR) utilisées dans les centres d'appels pour répondre aux demandes de certains utilisateurs.
- Applications d'assistant personnel telles que : OK Google, Siri, Cortana et Alexa.

La NLP implique l'application d'algorithmes pour identifier et extraire les règles du langage naturel de sorte que les données des langages non structurées soient converties en une forme que les ordinateurs peuvent comprendre.
Lorsque le texte est fourni, l'ordinateur utilisera des algorithmes pour extraire le sens associé à chaque phrase et en collecter les données essentielles.
Parfois, l'ordinateur peut ne pas bien comprendre le sens d'une phrase, ce qui conduit à des résultats imprévus.
L'analyse syntaxique et l'analyse sémantique sont les principales techniques utilisées pour réaliser les tâches de traitement du langage naturel.

Ainsi, NLP procède par plusieurs étapes afin de garantir un traitement précis :

1.1 - Enlever la ponctuation

La ponctuation peut fournir un contexte grammatical à une phrase qui soutient notre compréhension. Mais pour notre vectoriser qui compte le nombre de mots et non le contexte, il n'ajoute pas de valeur, nous supprimons donc tous les caractères spéciaux.
Par exemple: Comment vas-tu? -> Comment vas-tu.

1.2 - Tokenization

Tokenization (création de jetons) sépare le texte en unités telles que des phrases ou des mots. Il donne une structure à un texte précédemment non structuré.
Par exemple: Fleur du mal ===> «Fleur», «du», «mal»

1.3 - Enlever les stopwords :

Les stopwords sont des mots courants et répétitifs qui apparaîtront probablement dans n'importe quel texte. Ils ne nous en disent pas beaucoup sur nos données, nous les supprimons donc.
Par exemple: les pommes ou les bananes sont délicieuses ===> pommes, bananes, délicieuses.

1.4 - Prétraitement des données: Stemming

Le Stemming permet de réduire un mot à sa forme radicale. Il est souvent judicieux de traiter les mots apparentés de la même manière. Il supprime les suffixes, comme «ing», «age», «ance»..... par une approche simple basée sur des règles.
Cela réduit le corps de mots mais souvent les mots réels sont négligés.
Exemple: équipement, équipage ===> équip
Remarque: certains moteurs de recherche traitent les mots avec le même radical comme des synonymes.

2 - Encoders

Les nombres sont plus efficaces que les lettres pour les machines, car ils prennent moins d'espace dans le stockage, consomment moins de performance et on peut effectuer des opérations mathématiques avec.
C'est pour cela que l'encodage est très important lorsqu'on fait du machine learning.
On utilise principalement deux encodeurs : "one hot encoder" et "multi label binarizer".
Le "one hot encoder" encode un type de donnée comme un mot en un nombre, par exemple politique équivaut à 1 et économie équivaut à 4.

Le “multi label binarizer” fait la même chose que le “one hot encoder” mais organise le résultat de sortie sous forme de sparse matrix. On va définir la sparse matrix dans ce qui suit.

3 - Sparse Matrix

La matrice creuse ou sparse matrix en anglais est une matrice où on représente tous les éléments dont on a besoin et on remplit les cases non utilisées par des zéros.
Pour bien illustrer ce concept, imaginons un article de sport et un article de politique et économie :
On peut représenter l’article de sport avec une matrice de taille (1,1) et remplir la case avec le nombre équivalent à sport, et représenter l’article de politique et économie avec une matrice (2,1), mais cela va nous donner deux matrices avec deux tailles différentes ce qui va nous causer des problèmes lors des calculs.
La solution est de représenter les 2 articles avec une matrice (n,1), ce qui va nous donner par exemple (1,0,0,...,0) pour sport et (0,1,1,...,0) pour politique et économie. Chaque case dans ce cas correspond à une catégorie et n est le nombre de catégories. À noter que ce format consomme moins d’espace à stocker car on ne stocke que les cases pleines et que les cases peuvent seulement avoir 0 ou 1 comme valeurs.

4 - normalisation :

La normalisation consiste à modifier toutes les valeurs de façon à ce que tous les nombres aient le même intervalle de valeurs possible. Cela assure que lors du calcul, une valeur n’affecte pas les résultats du calcul car elle est sur une plus grande “échelle”.

5 - Feature Engineering :

Le feature engineering traduit en français est l'ingénierie de feature. Cela veut dire extraire des features pour améliorer l’algorithme. Sauf qu’il n’y a pas de guide ou de loi pour cette extraction de données car cela est lié directement à l’algorithme utilisé et au domaine de notre recherche, dans notre cas le journalisme ou la linguistique par exemple.

Pour donner un exemple, des données qui sont liées à la relation entre les phrases seraient inutile pour notre programme et détrimental pour notre temps de calcul car notre algorithme ne prend pas en compte les relations entre les phrases.
En contrepartie, puisqu'on a statistique et économie dans nos catégories et qu'on sait qu'un article d'économie ou de statistiques contient probablement ou certainement des nombres, alors on extrait le nombre de nombres mentionnés dans l'article comme feature. Cette même feature serait par contre inutile dans la classification qui détermine si cet article est positif ou négatif.

Au final, nous avons décidé sur ces features pour un équilibre de vitesse de calcul et de résultats satisfaisant :

1. catégories : NLP, nombre de nombres, longueur de l'article.
2. positif, négatif ou neutre : NLP, catégories (résultats de la prédiction précédente), longueur de l'article.
3. Complexe ou simple : NLP, catégories (résultats de la prédiction précédente), longueur de l'article.

c - Entraînement de l'algorithme :

Comme mentionné précédemment, nos algorithmes d'apprentissage sont des algorithmes supervisés de classification.
Révisons les classifications que peut avoir un article :
1 - 34 catégories possibles.
2 - positif, négatif ou neutre.
3 - simple ou complexe.

La 3éme classification est la plus simple, car c'est une classification binaire.
La 2éme classification va demander un peu plus de travail car elle est multiclasse.
La 1ere classification cependant est une classification multi label car toutes les catégories peuvent être possibles ce qui veut dire 34 classifications pour chaque article.
On étudiera dans les sections à venir les solutions qu'on a utilisées.

Ces solutions ont été choisies afin de garantir les meilleurs résultats tout en gardant le temps de calcul à moins d'une minute sur un processeur i5 5ème génération.

1 - Gaussian Naive-Bayes

Gaussian Naïve-Bayes (GNB) est le classifieur qu'on a choisi pour les 3 cas.
La classification Naïve-Bayes est une classification bayésienne probabiliste, cela veut dire qu'on se base sur des probabilités de certains événements observés et non calculés.
Ces probabilités sont bien sûr observées depuis nos 150 articles initiaux.

Un exemple simple pour illustrer comment cet algorithme marche :

Si 40 articles ont l'émotion "négative" alors la probabilité que l'article soit "négatif" est de 40/150.
On suppose que l'article contient 2 mots : covid et mort.
Maintenant, on calcul la probabilité que l'article soit "négatif" et aie le mot covid, de même pour "mort".
Et on multiplie les 3 probabilités obtenues.
Si on répète la même chose pour "positif", on peut facilement observer que la probabilité que l'article soit "négatif" et plus grande qu'il ne soit "positif".

Le "Multinomial Naïve-Bayes" aurais était le meilleur choix sauf qu'il ne supporte que des valeurs discrètes, mais puisqu'on a des valeurs non discrètes dans nos features comme la longueur de l'article, on a été forcé d'utiliser le GNB qui supporte les valeurs discrètes et continues.
On l'utilise directement dans la classification simple ou complexe (3eme classification).

2 - OvR

Le One Versus Rest (OvR) consiste à classifier chacune des possibilités contre toutes les autres possibilités puis prendre le résultat le plus probable des 3.

Dans le cas du naïve-bayes gaussien, ce concept peut être appliqué par exemple dans le calcul du cas neutre simplement en additionnant les probabilités des articles positifs et négatifs car leurs événements sont mutuellement exclusifs.

$$P(Y \text{ or } Z) = P(Y) + P(Z)$$

OVR est utilisée avec GNB dans la classification de positif complexe et neutre (2eme).

3 - Classifiers Chain

On a déjà mentionné que la classification des catégories est une classification multi label, cela veut dire que les événements ne sont pas mutuellement exclusifs. Si on applique le OvR dans ce cas, en plus d'avoir à calculer 34 classifications, on doit aussi prendre en compte la probabilité de l'événement de catégories en même temps car le calcul de l'addition de cette probabilité, je rappelle, est :

$$P(Y \text{ or } Z) = P(Y) + P(Z) - P(Y \text{ and } Z)$$

Ce calcul n'est pas réalisable car je rappelle qu'on possède seulement 150 articles, et 34 catégories.
Une autre approche serait de calculer chaque catégorie séparément des autres, on prend en compte dans ce cas seulement deux événements : la probabilité que le mot est utilisé et la probabilité que le mot n'est pas utilisé.
Cette approche s'appelle "Binary relevance" dans le cas du "multi label classification", mais elle ne prend pas en compte les relations entre les classes qu'on a, par exemple économie et statistique qui ont clairement une relation.
C'est pour cela qu'on utilise l'approche "Classifier Chain" (CC), une variante de "Binary relevance" qui combine la vitesse de calcul de cette dernière et prend en compte les relations entre classes.

Cette approche est utilisée avec GNB dans la classification de catégories (1ere).

ii - Web Technology Stack

a - Base de Données :

1 - Présentation de Django

Pour stocker nos données d'une manière simple et sécurisée, nous nous sommes servis d'un système de gestion de bases données propre à django. Django est un framework Python de haut niveau, permettant un développement rapide de sites internet, sécurisés, et maintenables. Créé par des développeurs expérimentés, Django prend en charge la plupart des tracas du développement web, vous pouvez donc vous concentrer sur l'écriture de votre application sans avoir besoin de réinventer la roue. Il est gratuit, open source, possède une communauté active, une bonne documentation, et plusieurs options pour du support gratuit ou non. Django vous aide à écrire une application qui est :

Polyvalente : Django peut être (et a été) utilisé pour créer presque tous les genres de sites — du gestionnaire de données aux wikis, jusqu'aux réseaux sociaux et aux sites d'actualités. Il peut fonctionner avec n'importe quelle infrastructure côté client, et peut renvoyer des données dans quasiment n'importe quel format (notamment HTML, RSS, JSON, XML, etc)

Sécurisé : Django aide les développeurs à éviter les erreurs de sécurité classique en fournissant une infrastructure conçue pour faire ce qu'il faut pour protéger les sites internet automatiquement. Il active par défaut la protection contre beaucoup de vulnérabilités, comme les injections SQL, le cross-site scripting, le cross-site request forgery et le clickjacking

Scalable : Django utilise une architecture composite "shared-nothing", c'est-à-dire que chaque composant de l'architecture est indépendant des autres, et peut ainsi être remplacé ou changé si besoin.

Maintenable : Les principes de design du code Django encouragent la création d'un code simple à maintenir et réutilisable. Il fait notamment appel à la philosophie du “Ne Vous Répétez Pas” (DRY pour Don't Repeat Yourself en anglais), afin d'éviter toute duplication superflue, réduisant la taille de votre code.

Portable : Django est écrit en Python, qui fonctionne sous diverses plateformes. Cela veut dire que vous ne serez plus contraint par une plateforme en particulier, et vous pourrez faire fonctionner vos applications sous autant de versions de Linux, Windows et Mac OS X que vous le souhaitez. De plus, Django est très bien supporté par plusieurs fournisseurs d'hébergement web, qui offrent souvent des infrastructures et de la documentation spécifiques pour héberger des sites Django.

2 - Base de données Psycopg :

Django prend officiellement en charge les bases de données suivantes :

- PostgreSQL
- MariaDB
- MySQL
- Oracle
- SQLite

Dans ce projet, nous avons adopté Psycopg comme système de base de données.
“Psycopg est l'adaptateur PostgreSQL le plus populaire pour le langage de programmation Python. Son cœur est une implémentation complète des spécifications Python DB API 2.0. Plusieurs extensions permettent d'accéder à de nombreuses fonctionnalités offertes par PostgreSQL.”

3 - Modèles

La création d’une base de données à l’aide de django nécessite la création d’un modèle.

Un modèle est la source unique et définitive d'informations sur vos données. Il contient les champs et les comportements essentiels des données que vous stockez. En général, chaque modèle correspond à une seule table de base de données.

- Chaque modèle est une classe Python qui sous-classe ***django.db.models.Model***.
- Chaque attribut du modèle représente un champ de base de données.
- Avec tout cela, Django vous offre une API d'accès à la base de données générée automatiquement.

Après la création de nos modèles, nous devons les ajouter à notre base de données. Dans un premier lieu, nous devons signaler à Django que nous venons de créer nos modèles. Ceci se fait à travers la commande *python manage.py makemigrations*

Django vient de nous préparer un fichier de migration que nous allons pouvoir appliquer dès maintenant à notre base de données à travers la commande suivante: *python manage.py migrate*.

b - Infrastructure

Comme expliqué précédemment, on utilisera nginx pour servir les fichiers statiques et django comme api.
Dans cette partie on expliquera l'architecture ainsi que nos solutions et le raisonnement derrière nos choix :
On a deux back-ends:

1. nginx qui sert les fichiers statiques créés par react.
2. l'api django qui traite les requêtes POST et renvoie les données appropriées.

On notera 3 systèmes importants : la recherche, la pagination et l'ouverture d'articles.

Dans les parties à venir on utilisera un exemple de cas d'utilisation pour illustrer notre système, vous trouverez ici une vidéo qui suit exactement les mêmes étapes. Je recommande fortement de regarder la vidéo en parallèle avec les parties à venir.

1. Recherche :

Notre implémentation de la recherche à deux objectifs majeurs : réduire l'utilisation de la base de données qui constitue notre plus grande consommation de mémoire et de puissance de calcul, et garantir une grande vitesse lors de l'utilisation de notre site en évitant de recharger la page à chaque recherche.
Voici l'exemple de cas d'utilisation étape par étape :

Premièrement, react envoie une requête POST à notre api django, qui contient les détails de la recherche, supposant que l'utilisateur écrit dans la barre de recherche "real madrid covid" et décoche la case welovebuzz.
Notre api reçoit la requête, exclu tous les articles provenant de welovebuzz, et décompose la recherche en mots : real, madrid et covid.
Il cherchera tous les mots contenant ces 3 mots clé et puisque covid est une des catégories qu'on a, il va inclure les articles qui sont classifié comme des articles de covid mais qui n'ont pas le mot clé covid dans leur textes (par exemple un article qui contient le mot coronavirus ou l'épidémie).
une fois qu'il possède la liste d'articles extraits par notre système de modèles expliqué précédemment, il crée une session et stocke tous les articles dans la base de donnée correspondantes en les codant avec le "pickle serializer", puis il renvoie la clé unique de session, le nombre de pages et les 12 premiers articles ainsi que leur données (titre, images, categories, source, url, id…)

Bien sûr on peut décocher toutes les catégories et sources ainsi que les articles positifs négatifs, neutre simples et complexes pour ne pas les afficher, ou bien afficher seulement les articles qui valident ces critères.

2. Pagination :

Une fois notre site react reçoit ces données, ils les stockent tous dans la mémoire de l'appareil, puis React-Dom commence à modifier le DOM du navigateur en utilisant ces données avec les actions suivantes :

- Il affiche la phrase de recherche ainsi que la page actuelle dans la "breadcrumb" dans une barre en haut.
- Il affiche le nombre de pages dans la barre de pagination en bas.
- Il affiche les articles dans le corps de la page soit par groupe de 3 articles par ligne (grands écrans/écrans moyen) ou groupes de 2 (tablettes ou mobile en mode horizontal) ou 1 article par ligne (mobile en mode vertical).

Lorsque l'utilisateur clique sur une page, react envoie la page désirée ainsi que la clé de session, notre api donc à seulement besoin de fournir les 12 articles correspondant à cette page en multipliant 12 par le nombre de la page, sans avoir besoin de refaire la recherche ou d'ouvrir notre base de données primaire.
Ceci veut dire qu'on ne consomme presque aucune puissance de calcul lorsqu'on change de page, réduisant la pression sur notre serveur et api et sans avoir à recharger la page non plus.

3. Ouverture d'articles :

Si un utilisateur clique sur un article, alors on stocke toutes les données de la recherche dans une "backup" puis on prend les données de cet article déjà obtenu lors de notre requête de recherche ou de changement de page, et on les utilisent pour créer une URL unique créée par l'id (clé primaire) de l'article, ainsi qu'afficher une page unique créée à partir de ces données comme l'url de l'article qu'on utilise pour ouvrir une "iframe" (ou un onglet vers l'article si le site n'autorise pas une iframe dans ces règles d'utilisation) et la source qu'on utilise pour afficher une barre mentionnant la revue.

On affiche aussi un bouton “retourner à la liste” qui affiche la page précédente en utilisant les données stockées dans la “backup” à la place de la requête POST comme décrit dans les parties précédentes.

Cela veut dire que lors de toute cette partie, on n'a ni envoyé de requête à notre api, ni rechargé la page, ce qui résulte en un gain massif de performance.

Alternativement, vous pouvez directement entrer l’url unique de l’article et react va envoyer une requête à l’api pour demander seulement les données de cet article et les afficher normalement.

IV - DU PASSÉ À L’AVENIR

i - Difficultés et Bugs

a - responsivité

On savait depuis le début qu’un grand nombre d'utilisateurs vont utiliser notre solution principalement sur le mobile. cela voulait dire qu’on devrait développer notre solution sur le web pour le desktop en utilisant des technologies pertinentes au web comme react et sur mobile en utilisant des technologies appart pour Android et pour Ios et peut être même pour harmony os prévu de sortir en 2021, cela voulait dire qu’on devrait utiliser 3 technologie séparées pour le même but, et cela veut dire 3 fois plus de temps de développement en plus de garder le design fidèle pour les 3 plateformes majeures. C’est pour cela qu’on a opté pour une différente approche : utiliser le web sur les 3 différentes plateformes.
sauf que cela veut dire qu’on doit afficher la même page sur un écran de 1920 px, 1200 px, 600 px et 400 px de largeur soit environ plus de 4 fois la différence de taille entre le plus grand et le plus petit, tout en garantissant que le texte soit parfaitement lisible et les boutons facilement cliquable. Cela prend toujours du temps supplémentaire mais beaucoup moins car il faut avoir des connaissances en profondeur dans le CSS et JS au lieu de devoir apprendre d’autres technologies.

ii - Roadmap :

Ce projet ne se terminera pas ici, car on a plein de plans et d'idées qu'on veut implémenter dans ce projet. On citera quelques-unes de ces idées :

- la presse en langues arabes : ce sera probablement la partie la plus difficile à cause de plusieurs complication, comme l'alphabet et le sens d'écriture différent, ainsi que plein d'autres problèmes relié à la complexité de la langue arabe comparée au langues latines ainsi qu'au manque de développement de la NLP pour la langue arabe qu'on devra couvrir nous même.
- une application mobile et possiblement une application sur pc et MacOs.
- une refonte du design du site : le design actuel du site web est temporaire.
- publier et monétiser la plateforme.
- une rubrique pour femmes et autres domaines spécifiques.
- la possibilité aux journalistes non affiliés de publier directement sur notre plateforme.
- auto-ML : permettre à l'algorithme de s'auto-former et de se développer au cours du temps avec les algorithmes qu'il collecte sans intervention humaine.

V - CONCLUSION :

Tout d'abord, ce projet nous a permis d'appliquer les connaissances que nous avons acquises durant cette année scolaire, telles que le développement personnel, les différents langages de programmation et surtout la communication.

Grâce à celui-ci chaque membre de l'équipe a pu renforcer ses connaissances mais aussi apporter aux autres membres son savoir et ses compétences afin d'harmoniser l'efficacité de l'équipe.

A travers des méthodes de travail et des outils, ce projet nous a permis de nous immerger dans un univers professionnel.

Il est vrai que de développer un site web/application et respecter un cahier des charges rendent un projet intéressant et professionnel mais il y a aussi toutes les démarches qui ne sont pas visibles et qui rendent enrichissante une telle expérience : écouter l'opinion de chacun des membres de l'équipe, savoir communiquer et argumenter afin d'opter pour les meilleurs choix, s'organiser sur les plans personnels et collectifs, gérer les imprévus, respecter des délais pour ne pas gêner ses collègues et pour ne pas retarder tout le projet.

Le projet nous apporte donc à chacun une idée sur l'organisation dans le monde professionnel et qui permettra de nous adapter plus facilement lors de notre stage.

REMERCIEMENT :

On exprime toute notre reconnaissance et gratitude à l'administration et à l'ensemble du corps enseignant de l'ENSAM Casablanca pour leurs efforts à nous garantir la continuité et l'aboutissement du programme de cette année malgré les circonstances difficiles qu'on a vécu durant cette crise sanitaire.

Un grand merci à nos familles, nos amis et nos collègues pour leurs soutiens, leurs confiances ainsi que leurs conseils inconditionnels d'encouragement qui nous ont poussés de toujours persévérer durant toute cette période pour qu'on parvienne à la finalisation de notre travail.

VI - WEBOGRAPHIE :

- ➔ https://fr.wikipedia.org/wiki/Beautiful_Soup
- ➔ https://fr.wikipedia.org/wiki/Attente_active
- ➔ https://fr.wikipedia.org/wiki/Traitement_automatique_des_langues
- ➔ https://becominghuman.ai/a-simple-introduction-to-natural-language-processing-ea66a1747b32
- ➔ https://towardsdatascience.com/natural-language-processing-nlp-for-machine-learning-d44498845d5b
- ➔ https://developer.mozilla.org/fr/docs/Learn/Server-side/Django/Introduction
- ➔ https://www.psycopg.org/
- ➔ https://docs.djangoproject.com/fr/3.1/topics/db/models/