Universite de Montpellier
Rapport de projet

Claims checking

Belkassim BOUZIDI
Chakib ELHOUITI
Massili KEZZOUL
Abdelkader NEDJARI
Ramzi ZEROUAL

Encadrant :
M^r Konstantin Todorov

10 mai 2020

Remerciements

Tout d'abord nous souhaitons adresser nos remerciements au corps professoral et administratif de la faculté des sciences de Montpellier qui déploient des efforts pour assurer à leurs étudiants une formation actualisée.

En second lieu, nous tenons à remercier notre encadrant Mr Konstantin Todorov pour ses précieux conseils et son aide durant toute la période du travail.

Nos vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à notre projet en acceptant d'examiner notre travail.

Nous remercions Mr Yahia Zeroual pour sa relecture attentive de ce rapport.

Table des matières

Chapitre 1

Organisation du projet

1.1 Méthodes d'organisation

Afin de mener à bien le développement du projet, nous avons décidé de travailler un maximum de temps ensemble et de manière très régulière. Nous nous sommes réunis trois à quatre fois par semaine, en vue de faire le point sur l'avancement du projet et de définir les objectifs restants à atteindre.

Ainsi, selon l'état de progression de la conception, nous réalisâmes les tâches en retard durant le week-end pour ne pas cumuler de retard et respecter l'intégralité du cahier de charges.

Toutes les semaines, nous nous sommes réunis avec notre encadrant, M^{r} Konstantin Todorov. Lors de ces réunions de mises au point relatifs au projet, nous furent prodigués, cela nous a permis de bénéficier de précieux conseils.

1.2 Découpage du projet

Nous avons découpé la réalisation du projet en trois grandes phases :

1.2.1 Phase de modélisation

Durant cette étape, nous nous sommes réunis pour définir les fonctionnalités demandées par le projet. Notamment séparer les fonctionnalités importantes de celles moins importantes. Nous avons également choisi les outils de travail collaboratifs et les principales technologies utilisées, ainsi qu'une première modélisation du projet.

1.2.2 Phase de développement

Durant cette phase, nous avons commencé à implémenter les différentes fonctionnalités que nous avons modélisées lors de l'étape précédente, tout en améliorant la modélisation au fur et à mesure de l'avancement de notre projet. Nous avons notamment réalisé des tests pour les différents modules afin de s'assurer de leur bon fonctionnement.

1.2.3 Finalisation du projet

Cette étape a consisté en la réalisation des tests finaux afin de s'assurer que les différents scripts fonctionnent en toute circonstance et éventuellement corriger les bogues qui peuvent apparaître.

1.3 Outils de collaboration

Afin de s'organiser, nous avons décidé d'utiliser Git à travers le serveur GitLab hébergé par le service informatique de la faculté. En effet le logiciel libre Git a facilité grandement la collaboration entre nous. Le serveur GitLab quant à lui est fourni gratuitement par le service informatique de la faculté.

En ce qui concerne la rédaction de ce rapport, nous avons utilisé LaTeX, système de composition de documents créé par Leslie Lamport, pour faciliter la rédaction à plusieurs.

Schéma 1.1 – Logo du GitLab

Schéma 1.2 – Logo de Latex

Chapitre 2

Introduction au sujet

2.1 Fact-checking

2.1.1 Présentation du principe de fact-checking

Le fact-checking ou le vérification des faits, est une technique consistant à vérifier la véracité des faits et l'exactitude des chiffres présentés dans les médias, les différents réseaux sociaux, les blogs, etc... Cette notion est apparue aux États-Unis dans les années 1990. Elle a été mise en pratique par des journalistes d'investigation dans le cadre de leur profession, la méthode s'est démocratisée grâce à des logiciels aidant les particuliers à vérifier les faits.

L'entrée officielle du fact-checking en France date de 1995, quand est créée l'association Acrimed, qui se présente comme « l'observatoire des médias ».

En vue de la prolifération très rapide des informations, il devient de plus en plus important de s'assurer de la véracité des informations qui se trouvent partout sur Internet et autres médias. Tant du point de vue de la société que de celui de la recherche. De nombreuses approches récentes dans diverses communautés scientifiques portent sur des problèmes tels que la vérification des faits, la détection de la pertinence ou de point de vue des documents par rapport à des revendications particulières.

2.1.2 Présentation de ClaimsKG

Le LIRMM [1] en collaboration avec 2 équipes allemandes (L3S Hannover et l'institut de sciences sociologiques GESIS à Cologne) a construit et mis à disposition la base de connaissance ClaimsKG [2] qui regroupe les informations et méta-données provenant d'un grand nombre de sites journalistiques internationaux de fact checking, tels que Politifact [3] ou Snopes [4]. ClaimsKG est un graphe de connaissances d'assertions annotées et liées qui facilite la création de requêtes structurées sur les assertions, leurs valeurs de vérité (True, Mostly False, etc...), leurs auteurs, date de publication, etc... ClaimsKG est généré par un pipeline entièrement automatisé qui collecte des assertions et des métadonnées à partir des sites de fact-checking, il transforme les données en graphes de connaissances selon un modèle établi, et annote les assertions avec des entités DBpedia (Wikipedia). La base actuelle comprend plus de 32 000 assertions publiées depuis 1996 et est mise à jour régulièrement.

1. Le Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier
2. https://github.com/claimskg
3. https://www.politifact.com/
4. https://www.snopes.com/

2.1.3 Travail à réaliser

Le sujet du TER consiste en l'enrichissement de cette base de connaissances avec des nouvelles données provenant des sites web suivants :

Fatabyyano [5] est un site jordanien en arabe. Fatabyyano (terme en arabe qui veut dire "Alors montrez-le") est la première et la seule plateforme arabe certifiée par l'IFCN [6] ;

Vishvas.news [7] est un site Internet de vérification des faits multilingue qui s'engage à combattre la désinformation et les informations erronées.

Le but du TER sera d'identifier les assertions individuelles dans chaque histoire, ainsi que leur label de véracité et par la suite identifier les relations entre elles, ainsi que les relations entre ces histoires. Les données produites par ce projet seront intégrées à la base de connaissance ClaimsKG.

La principale difficulté qu'on s'attend à rencontrer est la gestion des différentes langues proposée par ces sites web. Notamment dans l'identification des différentes relations entre les assertions. En effet afin de reconnaitre les différents mots-clés du sujet traité par les articles, on utilise habituellement TAGME, un puissant outil de reconnaissance d'entités nommées dans un texte. Les langues utilisées par ces sites web ne sont pas prises en charge par cet outil. Il s'agit donc de trouver une alternative afin de reconnaitre ses différentes assertions. Ce problème ainsi que sa résolution seront détaillés plus tard dans ce rapport.

2.2 Technologies utilisées

Nous avons implémenter l'application en Python, choix qui s'impose de lui même car couplé à la bibliothèque BeautifulSoup, il devient très facile de faire du web scraping [8].

En ce qui concerne la traduction, nous avons choisi d'utiliser Yandex Translator [9]. En effet en vue de la taille des données à traduire, nous n'avons pas pu utiliser l'API de traduction de Google puisque cette dernière devient payante à partir d'un certain nombre de caractères.

Enfin, la reconnaissance des différents entités concernées est faite grâce à l'outil TAGME [10] a travers son API python [11].

Schéma 2.1 – Logo de Python

Schéma 2.2 – Logo de BeautifulSoup

Schéma 2.3 – Logo de Yandex

5. https://fatabyyano.net/
6. International Fact-Checking Network : https://www.poynter.org/ifcn/
7. https://www.vishvasnews.com/
8. Le web scraping est une technique d'extraction du contenu de sites Web, via un script ou un programme
9. https://translate.yandex.com/
10. https://tagme.d4science.org/tagme/
11. Lien Github vers l'api Python de TAGME : https://github.com/marcocor/tagme-python

Chapitre 3

Conception et implémentation du projet

3.1 Conception et Modélisation

3.1.1 Analyse de ClaimKG

Lors de cette première phase, le plus important a été de comprendre et s'imprégner du code et de la structure déjà mis en place et mis à notre disposition (Claimskg), comprendre les outils utilisés ainsi que la structure déjà établie.

La structure

Comme le projet est d'une envergure immense, bien comprendre la structure était primordiale afin de respecter un maximum les outils utilisés. Notre encadrant nous à mis a disposition un écrit (que vous trouverez dans le dossier autre/ sous le nom de ClaimsKG.pdf) qui explique la structure globale du projet dont voilà ci-dessous un résumé.

Schéma 3.1 – ClaimKG

Comme vous pouvez le voir sur le schéma, ClaimsKG est découpé en plusieurs phases. La première consiste à extraire les données brutes à partir de site web de fact-checking et ensuite les structurer afin de produire un fichier CSV. La deuxième phase quant à elle, consiste à faire passer ce fichier CSV dans un générateur de graphes de connaissances [1]. Dans notre cas, on s'occupe de la première phase, c'est-à-dire, extraire les données brutes du site web afin de produire le fichier CSV.

Le fichier CSV en question contient l'ensemble des faits traités par un site web. Chaque fait est passé à un script de scraping afin d'extraite chaque information et la structurer. La structure d'un fait est réalisée selon un modèle précis dont voici une partie :

rating value : La valeur de véracité du fait ;

1. Knowledge Graph Generator

creativeWork author name : Auteur de la claim ;
creativeWork datePublished : Date de publication de la claim ;
claimReview author : Auteur de l'article/analyse de la claim ;
extra entities claimReview : Les différentes entités présentes dans la review.

Vous trouverez ci-dessous la structuration complète.

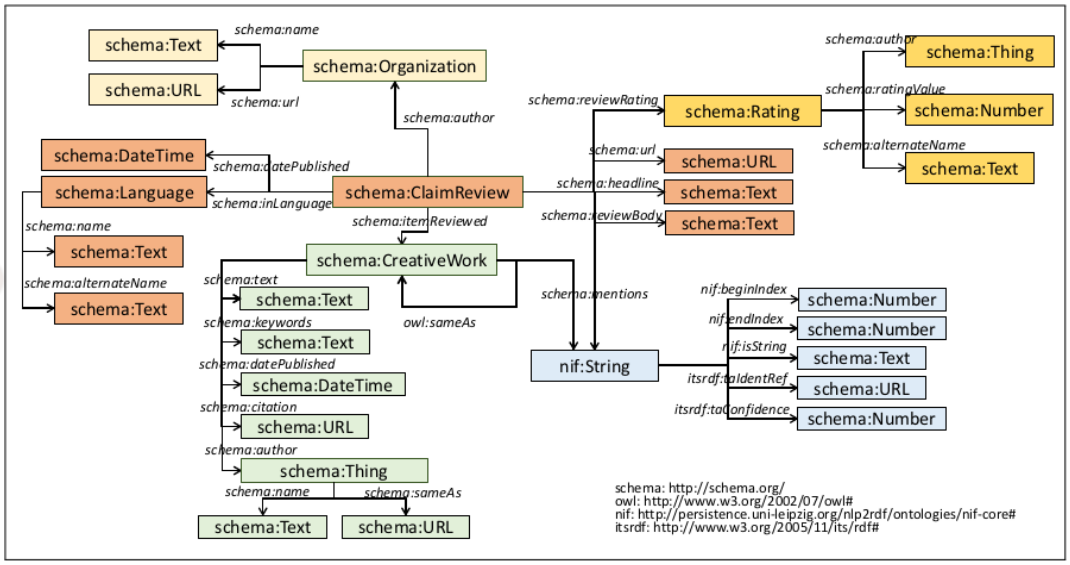

Schéma 3.2 – Structuration complète d'un fait

L'implémentation

Après la partie théorique de l'analyse nous avons commencer à regarder les implémentations déjà faite [2]. Nous avons analysé les classes principales comme celle de « Claim.py », puis comprendre comment l'exécution de l'extraction ce fessait avec « __init__.py ». nous avons ensuite analysé les différents scripts d'extraction de sites de fact-cheking tel que « africacheck.py ».

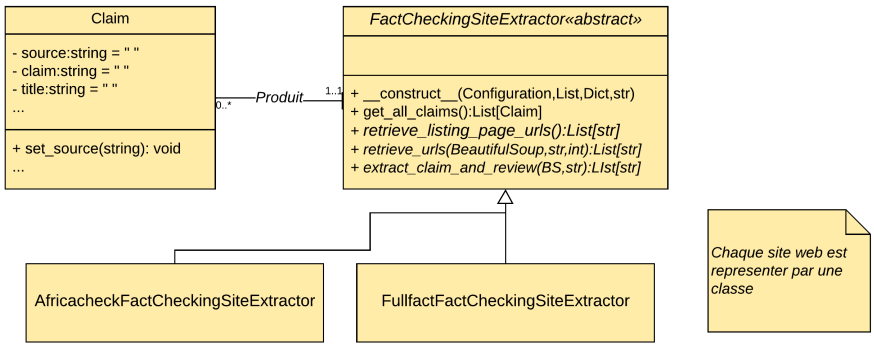

Schéma 3.3 – Structuration de claimsKG-extractor

2. Vous trouverez cette implémentation à cette addresse : https://github.com/claimskg/claimskg-extractor

3.1.2 Conception

Dans cette phase, nous allons présenter les étapes de conception que nous avons suivies pour bien implémenter notre application. La première étape a été de concevoir quelques diagrammes UML utiles à la bonne compréhension du projet.

Diagrammes UML

Lors de la phase de la modélisation, nous avons vu qu'il était primordial de concevoir un diagramme de classe pour connaitre la manière dont les classes interagissent. Comme la figure suivante le montre, des classes ont été rajoutées par rapport aux classes déjà développées.

Dans les sous-classes de FactCheckingSiteExtractor, la méthode "extract_claim_and_review" a été implémentée pour transformer une page donnée en paramètre en une instance de claim. La méthode "get_all_claims" quant à elle, extrait les liens des articles du site web. Les données, une fois extraites seront stockées dans un fichier CSV où chaque ligne représente un article et chaque colonne les informations le concernant.

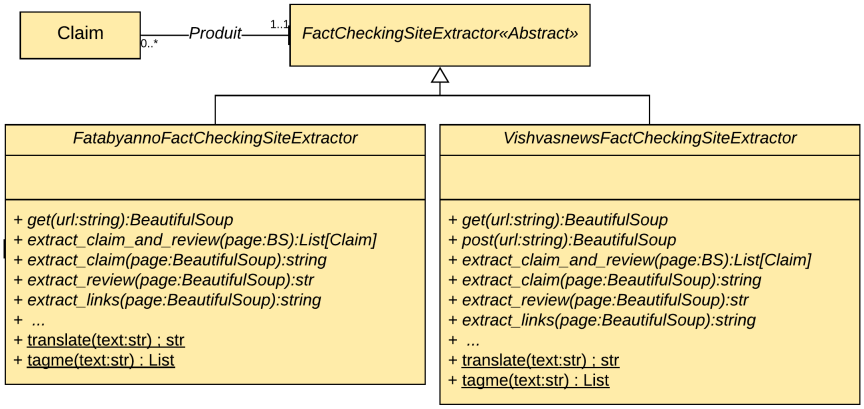

Schéma 3.4 – Structuration de claimsKG-extractor

Le déroulement de l'extraction des données a été quant à elle modélisée par un diagramme de séquence. La figure ci-dessous représente le fonctionnement de l'extraction du site Fatabayyano.

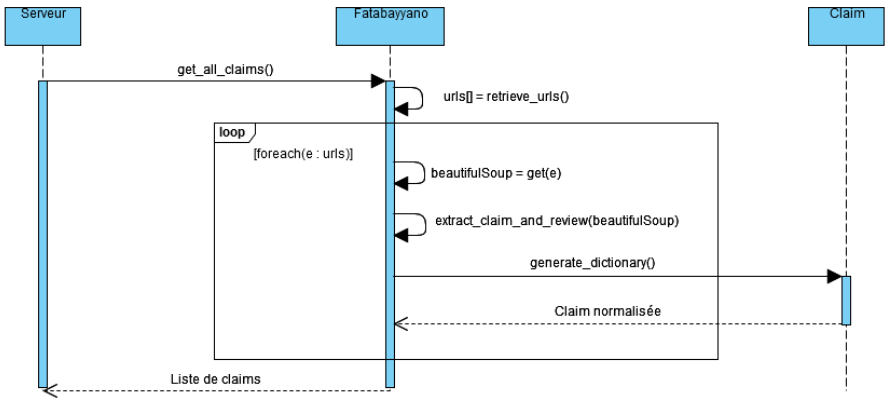

Schéma 3.5 – Structuration de claimsKG-extractor

Représentation des données

La deuxième étape consistait à trouver un site web respectant les critères de fact-cheking et qui nous permet d'extraire les données essentielles (la véracité, les claims écrites, un auteur, etc...) au noyau du projet (ClaimsKG). Pour apporter un maximum de diversité ainsi que notre contribution personnelle, nous avons décidé, avec l'accord de notre encadrant, d'extraire les données d'un site d'une langue qui n'est pas encore présente dans le projet. Vu que tous les étudiants de ce projet sont arabophones, la langue arabe était un choix logique. Le seul site officiel de fact-cheking en arabe est : https ://fatabyyano.net/.

Nous avons donc analysé la structure du site web puis commencé son implémentation. L'étape suivante fût de trouver comment faire une "named entity recognition" sur les mêmes principes que celles déjà présentées (TagMe qui renvoie vers des liens wikipedia).

Le second site web sur lequel nous avons travaillé est https ://www.vishvasnews.com/, il regroupe 11 langues différentes : L'anglais, Hindi, Pendjabi, Ourdou, Bengali, Tamoul, Malayalam, Goudjerati, Télougou, Marathi et L'odia , nous avons analysé la structure du site puis extrait ses données.

3.2 Implémentation

Tout comme la partie conception, nous avons décidé de répartie l'implémentation du projet en X parties

Nous avons commencé par implémenter les classes que nous avons modélisées dans la partie conception, puis nous nous sommes concentrés sur la manière d'extraire les liens de chaque article. Pour cela, nous avons tout d'abord chercher dans le site où et comment les liens sont presentés en inspectant les pages web ainsi que leur code source. Une fois les liens trouvés et après avoir bien cerné la structure du site web, nous avons commencé à ecrire la méthode « get(url :string) » qui permet, à partir d'un lien URL de retourner une instance de la classe « BeautifulSoup ». Cette classe permet de garder en mémoire le code source d'une page web de façon structurée, et qui permet avec ses méthodes, de recherche et de manipulation. Elle facilite aussi grandement l'extraction des données. Une fois cette méthode achévé, on a pu commencer à ecrire la méthode « retrieve_url(page :BeautifulSoup) » qui recupere l'ensemble des liens des articles.

L'ensemble des liens maintenant récupéré, nous avons implémenté les méthodes d'extraction - « extact_claim(page :BeautifulSoup) »,« extract_date(page :BeautifulSoup) »...etc - en analysant la position de chaque information une à une dans la structure du code source.

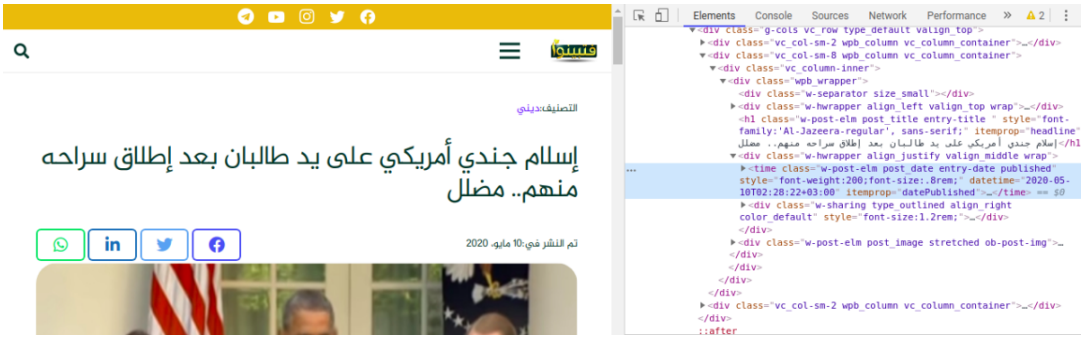

Schéma 3.6 – Extraction de la date du site web Fatabyyano

Ensuite une fois les données extraite du site, nous avons réfléchi à un moyen d'extraire les entités présentent dans la claim et dans la review de l'article. Pour se faire, nous avons cherché une alternative à TAGME pour la lague arabe (pour le site web Fatabyyano), après plusieurs recherches et tests, nous n'avons trouvé aucune solution plausible pour le remplacer. Nous avons opté pour une traduction de l'arabe vers l'anglais en utilisant une API de Yandex, pour ensuite

Schéma 3.7 – Extraction de la véracité du site web Vishvas

pouvoir utiliser TAGME sur l'article traduit.

Pour finir, après avoir implémenté un script en utilisant notre conception, nous avons pu générer une première version du fichier CSV. par la suite, nous nous sommes retrouvés dans l'obligation de le modifier pour qu'il soit en adéquation avec l'architecture du fichier CSV du projet principal (ClaimsKG).

Chapitre 4

Analyse des résultats

4.1 Résultats

Comme premier résultat du site fatabayyano nous avons obtenu un fichier csv contenant l'url de la claim, le rating, la traduction la claim, la date, le titre, tous les liens présents sur la page et les tags ; le tout en langue arabe.

Url	rating	rating_trans	claim	date	tags
https://fataby	صحيح	TRUE	في ولاية راخين	2019-07-25	"الشرطة,بورما
https://fataby	صحيح	TRUE	في رجل مسلم	2019-07-25	"اكات,بورميين
https://fataby	صحيح	TRUE	اكان في بورما	2019-07-25	"بورما,تعذيب
https://fataby	صحيح	TRUE	تين غرب بورما	2019-07-25	"تدمير,تكشف
https://fataby	صحيح	TRUE	لهاداً في العالم	2019-07-25	"هادا,الروهينجيا
https://fataby	صحيح	TRUE	يوك عن العمل	2019-07-03	"وم,علوم,طبي
https://fataby	صحيح	TRUE	يقة أم إشاعة؟	2019-05-15	"وم,علوم,طبي
https://fataby	صحيح	TRUE	يقة أم مفبرك؟	2019-05-06	"وم,علوم,طبي
https://fataby	صحيح	TRUE	لثانية في 2019	2019-04-14	"وم,علوم,طبي
https://fataby	صحيح	TRUE	أبوظبي و دبي	2019-04-03	"أخبار_حقيقية,
https://fataby	صحيح	TRUE	وسط كوبنهاغن	2019-03-26	"دنماركيون#,ة
https://fataby	صحيح	TRUE	خدمي فيسبوك	2019-03-23	"وم,علوم,طبي
https://fataby	صحيح	TRUE	خلال بث مباشر	2019-03-16	"Fraser_Anni
https://fataby	صحيح	TRUE	تقرير فتبينوا ..	2019-03-15	"newzealand,
https://fataby	صحيح	TRUE	مل حول العالم	2019-03-13	"وم,علوم,طبي
https://fataby	صحيح	TRUE	لادش المجاورة	2019-02-27	"بنغلادش,حقيقة
https://fataby	صحيح	TRUE	قلب ولا نبض ؟	2019-02-23	"وم,علوم,طبي
https://fataby	صحيح	TRUE	دين الإسلامي ؟	2019-02-15	"Amazon,مات
https://fataby	صحيح	TRUE	يقة ام إشاعة ؟	2019-02-06	"Islam,هولندا,
https://fataby	صحيح	TRUE	ر ضد الإمارات؟	2019-01-11	"حطيرة_حيوان
https://fataby	صحيح	TRUE	ي سماء كوينز؟	2018-12-29	"ت,ضوء#,أزرق
https://fataby	صحيح	TRUE	بعيون حمراء؟	2018-09-01	"وم,علوم,طبي
https://fataby	صحيح	TRUE	طاولة الجزار؟	2018-08-31	"وم,علوم,طبي
https://fataby	صحيح	TRUE	معدني منصهر؟	2018-08-19	"وم,علوم,طبي
https://fataby	صحيح	TRUE	ي الحرم المكي	2018-06-09	"وم,علوم,طبي
https://fataby	صحيح	TRUE	يداخل شجرة؟	2018-05-11	"وم,علوم,طبي
https://fataby	صحيح	TRUE	يقة أم خرافة؟	2018-03-01	"وم,علوم,طبي

Schéma 4.1 – premier fichier csv obtenu

Puis après avoir appliqué le processus de traduction sur la claim et la review et après obtention des entités présentes dedans, deux autres colonnes sont ajoutées contenant les résultats du tagme du projet initial, ces résultats sont les noms attribués par la base de données wikipedia de cette entité.

Les résultats du second site web "vishvasnews" sont aussi regroupés dans un fichier CSV, qui est généré par le programme du projet initial compte tenu du fait qu'une des langues extraites est l'anglais. Les colonnes contiennent les mêmes informations que le site précédent. Le tout est en anglais, Hindi, Pendjabi, Ourdou, Bengali, Tamoul, Malayalam, Goudjerati, Télougou, Marathi puis L'odia puis par catégorie.

Claim_url	Claim	title	claim_author	links	date	tags	authors	rating_value	claim_entitie	review_entit
https://fataby	"ن.. ما الحقيقة؟"	"ن.. ما الحقيقة؟"		https://fataby	2020-03-25	س,فيروس,هانتا	fatabyyano	صحيح	[{"id": 783925	[{"id": 589774
https://fataby	"ن.. ما الحقيقة؟"	"ن.. ما الحقيقة؟"		https://www.f	2020-03-18	ن,كورونا,ووهان	fatabyyano	صحيح	[{"id": 149109	[{"id": 82138, '
https://fataby	"، ما الحقيقة ؟"	"، ما الحقيقة ؟"		https://twitte	2020-03-17	_الطيور,فيروس	fatabyyano	صحيح	[{"id": 103410	[{"id": 322141
https://fataby	"ء.. ما الحقيقة؟"	"ء.. ما الحقيقة؟"		https://www.f	2020-03-06	لسعودية,كورونا	fatabyyano	صحيح	[{"id": 645068	[{"id": 645068
https://fataby	"قا..ما الحقيقة؟"	"قا..ما الحقيقة؟"		https://fataby	2020-02-24	ب,دم,الفلامينغو	fatabyyano	صحيح	[{"id": 165928	[{"id": 201483
https://fataby	"ن هاكرز إيراني"	"ن هاكرز إيراني"		https://www.f	2020-02-01	رز,وزارة الصحة	fatabyyano	صحيح	[{"id": 215107	[{"id": 247154
https://fataby	"ا حقيقة ذلك ؟"	"ا حقيقة ذلك ؟"		https://web.fa	2020-01-24	,فيروس_كورونا	fatabyyano	صحيح	[{"id": 371324	[{"id": 371324
https://fataby	"ه!...خبر صحيح"	"ه!...خبر صحيح"		https://www.f	2020-01-16	ي,يوسف_ديدات	fatabyyano	صحيح	[{"id": 228525	[{"id": 190758
https://fataby	"راليا عام 2019"	"راليا عام 2019"		https://www.c	2020-01-08	ي,حرائق,غابات	fatabyyano	صحيح	[{"id": 208277	[{"id": 56106, '
https://fataby	"معلومة صحيحة"	"معلومة صحيحة"		https://www.f	2020-01-08	تباس_الحراري	fatabyyano	صحيح	[{"id": 396275	[{"id": 468926
https://fataby	"ن.. ما الحقيقة؟"	"ن.. ما الحقيقة؟"		https://fataby	2020-03-25	س,فيروس,هانتا	fatabyyano	صحيح	[{"id": 783925	[{"id": 589774
https://fataby	"ن.. ما الحقيقة؟"	"ن.. ما الحقيقة؟"		https://www.f	2020-03-18	ن,كورونا,ووهان	fatabyyano	صحيح	[{"id": 149109	[{"id": 82138, '
https://fataby	"، ما الحقيقة ؟"	"، ما الحقيقة ؟"		https://twitte	2020-03-17	_الطيور,فيروس	fatabyyano	صحيح	[{"id": 103410	[{"id": 322141
https://fataby	"ء.. ما الحقيقة؟"	"ء.. ما الحقيقة؟"		https://www.f	2020-03-06	لسعودية,كورونا	fatabyyano	صحيح	[{"id": 645068	[{"id": 645068
https://fataby	"قا..ما الحقيقة؟"	"قا..ما الحقيقة؟"		https://fataby	2020-02-24	ب,دم,الفلامينغو	fatabyyano	صحيح	[{"id": 165928	[{"id": 201483
https://fataby	"ن هاكرز إيراني"	"ن هاكرز إيراني"		https://www.f	2020-02-01	رز,وزارة الصحة	fatabyyano	صحيح	[{"id": 215107	[{"id": 247154
https://fataby	"ا حقيقة ذلك ؟"	"ا حقيقة ذلك ؟"		https://web.fa	2020-01-24	,فيروس_كورونا	fatabyyano	صحيح	[{"id": 371324	[{"id": 371324
https://fataby	"ه!...خبر صحيح"	"ه!...خبر صحيح"		https://www.f	2020-01-16	ي,يوسف_ديدات	fatabyyano	صحيح	[{"id": 228525	[{"id": 190758
https://fataby	"راليا عام 2019"	"راليا عام 2019"		https://www.c	2020-01-08	ي,حرائق,غابات	fatabyyano	صحيح	[{"id": 208277	[{"id": 56106, '
https://fataby	"معلومة صحيحة"	"معلومة صحيحة"		https://www.f	2020-01-08	تباس_الحراري	fatabyyano	صحيح	[{"id": 396275	[{"id": 468926
https://fataby	"؟ تقرير مفصل"	"؟ تقرير مفصل"		https://www.g	2019-12-26	شمس,الكسوف	fatabyyano	صحيح	[{"id": 216958	[{"id": 502488
https://fataby	"طحة!....حقيقة"	"طحة!....حقيقة"		https://www.y	2019-11-27		fatabyyano	صحيح	[{"id": 8221, "t	[{"id": 8221, "t
https://fataby	"مقارب!..حقيقة"	"مقارب!..حقيقة"		https://www.v	2019-10-12	,العقارب,تدخين	fatabyyano	صحيح	[{"id": 271588	[{"id": 729317
https://fataby	"ي ثم بالسيف"	"ي ثم بالسيف"		https://www.y	2019-07-25		fatabyyano	صحيح	[{"id": 21201, '	[{"id": 603791
https://fataby	"ي ولاية راخين"	"ي ولاية راخين"		https://www.y	2019-07-25	ة,الشرطة,بورما	fatabyyano	صحيح	[{"id": 19457, '	[{"id": 887716
https://fataby	"ي رجل مسلم"	"ي رجل مسلم"		https://www.c	2019-07-25	تباكات,بورميين	fatabyyano	صحيح	[{"id": 992679	[{"id": 326752

Schéma 4.2 – fichier csv final de fatabayyano

Rating_rating	Rating_worst	Rating_best	Rating_altern	Creative Wor	Creative Wor	Creative Wor	Claim Review	Claim Review	Claim Review	Claim Review	Claim Review	Claim Review	Claim Rev
False			None	"FB Page: Dru			"['Urvashi Kap		https://www.v	"Drinking cold	2019-11-8	"['Urvashi Kap	
False			None	"FB User: Ragl			"['Urvashi Kap		https://www.v	"Nestle accep	2019-11-7	"['Urvashi Kap	
Misleading			None	"FB User: Jodi			"['Urvashi Kap		https://www.v	"Flu shots mal	2019-11-7	"['Urvashi Kap	
False			None	"FB User: Tim			"['Urvashi Kap		https://www.v	"Cream of Tar	2019-11-5	"['Urvashi Kap	
Misleading			None	"சுரேஷ் இசக்			"['Urvashi Kap		https://www.v	"Diabetes can	2019-11-4	"['Urvashi Kap	
Misleading			None	"FB Page-Patr			"['Urvashi Kap		https://www.v	"Cancer is not	2019-10-29	"['Urvashi Kap	
False			None	"FB User- Rog			"['Urvashi Kap		https://www.v	"Dr's say thos	2019-10-25	"['Urvashi Kap	
False			None	"Youtube char			"['Pallavi Mish		https://www.v	"Look at the t	2019-10-24	"['Pallavi Mish	
False			None	"FB User-Jay L			"['Urvashi Kap		https://www.v	"Miracle Mine	2019-10-19	"['Urvashi Kap	
Misleading			None	"Fb User- Jeff			"['Urvashi Kap		https://www.v	"pricking a ne	2019-10-18	"['Urvashi Kap	
False			None	"FB User-Oma			"['Urvashi Kap		https://www.v	"cockroach in	2019-9-26	"['Urvashi Kap	
False			None	"FB User-Jigya			"['Urvashi Kap		https://www.v	"This photo ca	2019-9-24	"['Urvashi Kap	
False			None	"FB User-Hake			"['Urvashi Kap		https://www.v	"The new-bor	2019-9-23	"['Urvashi Kap	
Misleading			None	"FB User-घरेलु			"['Urvashi Kap		https://www.v	"Sprouted wh	2019-9-23	"['Urvashi Kap	
False			None	"FB Page: Gha			"['Urvashi Kap		https://www.v	"Baba sucks ca	2019-9-14	"['Urvashi Kap	
Misleading			None	"FB User: Ugw			"['Urvashi Kap		https://www.v	"Ginger is up t	2019-9-10	"['Urvashi Kap	
Misleading			None	"FB User: Rajd			"['Urvashi Kap		https://www.v	"Pepaero cure	2019-9-6	"['Urvashi Kap	
Misleading			None	"Social media			"['Urvashi Kap		https://www.v	"Okra-soaked	2019-9-5	"['Urvashi Kap	
False			None	"FB User: Vnit			"['Urvashi Kap		https://www.v	"Noodles bein	2019-9-4	"['Urvashi Kap	
Misleading			None	"FB Page: San			"['Urvashi Kap		https://www.v	"Slaked lime c	2019-9-4	"['Urvashi Kap	
Misleading			None	"FB Page: Girl			"['Urvashi Kap		https://www.v	"Contraceptiv	2019-9-2	"['Urvashi Kap	
False			None	"FB Page: Pan			"['Urvashi Kap		https://www.v	"a drowned pe	2019-8-24	"['Urvashi Kap	
False			None	"FB User: Ben			"['Urvashi Kap		https://www.v	"This exercise	2019-8-7	"['Urvashi Kap	
False			None	"Whatsapp us			"['Urvashi Kap		https://www.v	"Knuckle Crac	2019-8-2	"['Urvashi Kap	
Misleading			None	"FB Page: Nili			"['Urvashi Kap		https://www.v	"Caripill cures	2019-7-31	"['Urvashi Kap	
False			None	"FB User: Sha			"['Urvashi Kap		https://www.v	"Paracetamol	2019-7-29	"['Urvashi Kap	

Schéma 4.3 – fichier csv de vishvasnews

4.2 Problèmes rencotrés

Le tout premier problème que nous avons rencontré a été lors de l'exécution des scripts d'extraction du projet ClaimsKg. En effet, la majorité des scripts que nous avons essayés ne marchaient plus, certains étaient anciens et n'étaient plus en adéquation avec les mises à jour, soit du projet en lui-même, soit du site web et cause de cela, les bases sur lesquelles nous devions prendre nos marques étaient à prendre avec des pincettes.

Le problème majeur suivant a été de trouver un moyen de faire une « named entity recognition » en langue arabe, car le programme " tagme " ne peut être utilisé sur l'arabe. Nous avons donc cherché des logiciels en open source qui pouvait nous aider à résoudre ce problème, malheureusement, aucun logiciel n'a été satisfaisant ; certains nous ont permis comme "Arabic NER" ou " NERAr "de faire des entity recognition, mais sans retourner les liens Wikipedia de ces derniers, d'autres comme " AiDA " qui semblaient être la solution ne fonctionnaient tout simplement pas. Nous avons donc décidé de traduire les claims et leurs reviews afin d'appliquer le programme "tagme", mais là aussi, les API de traduction ne marchaient pas correctement comme googletrans (googletrans est différente de google translate qui n'est pas gratuite) ou étaient limitées par jour comme celle que nous avons utilisé " Yandex ". Le souci avec cette api était que le nombre de caractère traduit quotidiennement était limités.

Le dernier gros problème rencontré quant à lui est le fait que trois des étudiants de notre groupe ont eu des soucis avec leurs machines - deux pc tombés en panne et un chargeur perdu - quelques jours après le début du confinement. Cette contrainte nous a empêché d'aller aussi loin dans la réalisation du projet et nous a retardé sur l'avancement global.

Chapitre 5

Bilan et conclusions

5.1 Perspective

Le scrapping des sites s'effectue parfaitement, aucune amélioration n'est nécessaire au script en lui-même, la seule perspective pour notre projet est la traduction des 10 langues présentes dans le deuxième site non prises en compte par le programme "tagme", deux idées peuvent être avancées :

Continuer à utiliser Yandex malgré ces capacités limitées tel qu'expliqué plus haut en changent uniquement le script de traduction du premier site web ou utiliser une autre API, de préférence Google Traduction qui est cependant payante, mais dont l'efficacité est sans conteste la meilleure.

5.2 Conclusion

Pour conclure, ce projet nous a permis d'apprendre beaucoup de nouvelles notions. Il a renforcé notre travail en groupe et le sens de la communication, de la synchronisation ainsi que de l'organisation. Il nous a aussi permis de découvrir les principes du web scrapping et l'utilisation de nouvelles API et bibliothèques.

Annexe A

Annexe

A.1 Code de la méthode *extract_claim_and_review*

```
    def extract_claim_and_review(self, parsed_claim_review_page: BeautifulSoup, url
    : str) -> List[Claim]:
        self.claim = self.extract_claim(parsed_claim_review_page)
        self.review = self.extract_review(parsed_claim_review_page)

        claim = Claim()
        claim.set_rating_value(
            self.extract_rating_value(parsed_claim_review_page))
        claim.set_alternate_name(FatabyyanoFactCheckingSiteExtractor.
    translate_rating_value(
            self.extract_rating_value(parsed_claim_review_page)))
        claim.set_source("fatabyyano")
        claim.set_author("fatabyyano")
        claim.setDatePublished(self.extract_date(parsed_claim_review_page))
        claim.set_claim(self.claim)
        claim.set_body(self.review)
        claim.set_refered_links(self.extract_links(parsed_claim_review_page))
        claim.set_title(self.extract_claim(parsed_claim_review_page))
        claim.set_date(self.extract_date(parsed_claim_review_page))
        claim.set_url(url)
        claim.set_tags(self.extract_tags(parsed_claim_review_page))
        # extract_entities returns two variables
        json_claim, json_body = self.extract_entities(self.claim, self.review)
        claim.set_claim_entities(json_claim)
        claim.set_body_entities(json_body)

        return [claim]
```

A.2 Code de la traduction

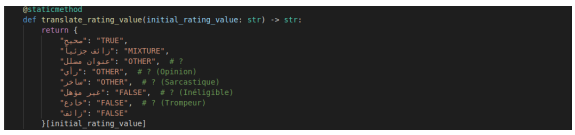

Schéma A.1 – Code traduction

```
    @staticmethod
    def translate(text: str) -> str:
        """
            :text: --> The text in arabic
```

```
            :return:  --> return a translation of :text: in english
        """
        if text == "":
            return ""
        self = FatabyyanoFactCheckingSiteExtractor
        yandexAPI = self.YANDEX_API_KEY
        yandex = YandexTranslate(yandexAPI)

        responses = []
        try:
            response = yandex.translate(text, 'ar-en')
            responses = [response]
            text_too_long = False
        except YandexTranslateException as e:
            if e.args == 'ERR_TEXT_TOO_LONG' or 'ERR_TEXT_TOO_LONG' in e.args:
                text_too_long = True
            else:
                print("Erreur API Yandex\nCode d'erreur : " + str(e.args))
                sys.exit(1)

        text_list = [text]

        while text_too_long:
            text_too_long = False
            try:
                text_list = self.cut_str(text_list)
            except ValueError:
                print("Erreur ")
                sys.exit(1)
            responses = []
            for t in text_list:
                try:
                    responses.append(yandex.translate(t, 'ar-en'))
                except YandexTranslateException:
                    text_too_long = True
                    continue

        text_list = []
        for r in responses:
            if int(r['code'] != 200):
                print(
                    "Erreur lors de la traduction\nCode de l'erreur : " + r['code'
    ])
                sys.exit(1)
            else:
                text_list.append(r['text'][0])

        return self.concat_str(text_list)

    @staticmethod
    def tagme(text):
        """
            :text:  --> The text in english after translation
            :return:  --> return a list of entities
        """
        if text == "":
            return []
        tagme.GCUBE_TOKEN = FatabyyanoFactCheckingSiteExtractor.TAGME_API_KEY
        return tagme.annotate(text)
```

A.3 Code de la méthode *retrieve_urls*

```
            :number_of_page:      --> number_of_page
            :return:              --> la liste des url de toutes les claims
        """
        links = []
        select_links = 'ul.listing li div.imagecontent h3 a'
```

```
        # links in the static page
        claims = parsed_listing_page.select(
            "div.ajax-data-load " + select_links)
        for link in claims:
            if link["href"]:
                links.append(link["href"])

        # for links loaded by AJAX
        r = re.compile(
            "https://www.vishvasnews.com/(.*)/(.*)[/]").match(listing_page_url)

        lang = r.group(1)
        categorie = r.group(2)

        url_ajax = "https://www.vishvasnews.com/wp-admin/admin-ajax.php"
        data = {
            'action': 'ajax_pagination',
            'query_vars': '{"category_name" : "' + categorie + '", "lang" : "' +
    lang + '"}',
            'page': 1,
            'loadPage': 'file-archive-posts-part'
        }

        response = self.post(url_ajax, data)

        while True:
            claims = response.select(select_links)
            for link in claims:
                if link['href']:
                    links.append(link['href'])

            if response.find("nav"):
                data['page'] = data['page'] + 1
                response = self.post(url_ajax, data)
                continue
            else:
                break

        return links
```

A.4 Code de la méthode *extract_links*

```
    def extract_links(self, parsed_claim_review_page: BeautifulSoup) -> str:
        links = []

        # extracting the main article body
        review_body = parsed_claim_review_page.select_one(
            "div.lhs-area")
        # making a clone
        review_body = copy.copy(review_body)

        # removing the social-media sahres links
        review_body.select_one('ul.social-icons-details').decompose()

        # removing authors  & tag  links
        b = False

        # ( > * ) ==> direct children in css
        for tag in review_body.select("> *"):
            if tag.get('class') and "reviews" in tag.get('class'):
                b = True
            if b:
                tag.decompose()
            else:
                continue
        # extracting links
        for link_tag in review_body.select('a'):
```

```
            if link_tag.has_attr('href'):
                links.append(link_tag['href'])

        return self.escape(str(links))
```